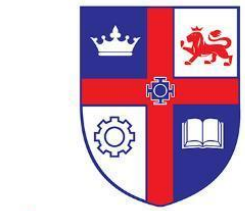
LONDON METROPOLITAN UNIVERSITY



islington college
(इस्लिङ्टन कलेज)

CS4001NI Programming

30% Individual Coursework

2022-23 Autumn

Student Name: Sujal Shrestha

London Met ID: 22068166

College ID: NP01CP4A220517

Group: L1C14

Assignment Due Date: Wednesday, May 10, 2023

Assignment Submission Date: Wednesday, May 10, 2023

I confirm that I understand my coursework needs to be submitted online via MySecondTeacher under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarde

Table of Contents

Table of Figure

Table of Tables

1. Introduction

GUI which stands for Graphical User Interface is a form of Interface which allows users to interact with devices like Laptops, PCs, Mobiles etc using various icons, menus and Indicators. In this modern and technologically advanced world, a GUI can make all the difference in enhancing user experience and making complex tasks easier to do. In this report, GUIs I have developed for Bankcard, Debit card, and Credit Card using Java language. These GUIs not only improve the functionality of these Bank Cards but also make them more accessible to users of all affiliated backgrounds.

The functionality for the adding, withdrawing, cancelling, Displaying make this GUI more enhance and user friendly. Certainly there are 3 GUIs in which Bank Card is the parent GUI. You must make an account in Bank Card, to operate with others GUIs . In Debit Card You can withdraw amount from your Account using correct Card Id and Pin code. More Likely, You can have a credit according to your account balance Amount for interest rate and grace period as well Cancel you Credit Using correct enrolment Id. These are some Features of my Function able GUIs.

This project is fully developed using java Languages and its various libraries. Moreover, I used Bluej tool for my project which amplify my working speed. This tool gave me advance feature and suggestions which makes me more efficient and well organized. Furthermore, I used draw.io for the class diagram which provide me easy assibilate and very quality facilities although being open source. All the shapes and diagram can be easily be developed and maintain as needed without any high level training.

2. Class Diagram

Class Diagram is the systematic way of presenting the class, attribute, constructor and method of the specific class. It is the type of static structure diagram in the Unified modelling language which shows the relation among the objects. It is helpful for the programmers another teams too. It can be used in the business perspective for modelling the system. It also help to show which kind of relation it holds with other class too. Notes are also can be attached with the class diagram in grey so can be used in understand. We can built two different type of class diagram single class diagram as well as multiple class diagram (keller, 2001).

BankGUI
-bankcard : Arraylist -lblTitle : JLabel -lblID : JLabel -txtID : JTextField -lblAccount : JLabel -txtAccount : JTextField -lblName : JLabel -txtName : JTextField -lblAmount : JLabel -txtAmount : JTextFeild -lblIssuer : JLabel -txtIssuer: JTextField -lblPin : JLabel -txtPin : JTextField -lblConfirmPin : JLabel -txtConfirmPin : JTextField -lblDate : JLabel -boxDD : JComboBox -boxMM : JComboBox -boxYY : JComboBox -btnCreate : JButton -btnDisplay : JButton -btnClear : JButton

-btnGoDebit : JButton
-btnGoCredit : JButton
-lblDebTitle : JLabel
-lblDebID : JLabel
-txtDebID : JTextField
-btnDebGo : JButton
-lblDebAccount : JLabel
-txtDebAccount : JTextField
-lblDebName : JLabel
-txtDebName : JTextField
-lblDebAmount : JLabel
-txtDebAmount : JTextFeild
-lblDebIssuer : JLabel
-txtDebIssuer: JTextField
-lblDebPin : JLabel
-txtDebPin : JTextField
-lblWithDraw : JLabel
-txtWIthDraw : JTextField
-lblDebDate : JLabel
-boxDebDD : JComboBox
-boxDebMM : JComboBox
-boxDebYY : JComboBox
-btnWIthDraw : JButton
-btnDebDisplay : JButton
-btnDebClear : JButton
-btnDebGoBank : JButton
-lblCreTitle : JLabel
-lblCreID : JLabel
-txtCreID : JTextField
-btnCreGo : JButton
-lblCreAccount : JLabel
-txtCreAccount : JTextField
-lblCreName : JLabel

-txtCreName : JTextField -lblCreAmount : JLabel -txtCreAmount : JTextFeild -lblCreIssuer : JLabel -txtCreIssuer: JTextField -lblCrePin : JLabel -txtCrePin : JTextField -lblCvc : JLabel -txtCvc : JTextField -lblLimit : JLabel -txtLimit : JTextField -lblInterest : JLabel -txtInterest : JTextField -lblGrace : JLabel -txtGrace : JTextField -lblExpiration : JLabel -boxCreDD : JComboBox -boxCreMM : JComboBox -boxCreYY : JComboBox -btnCreAdd : JButton -btnSet : JButton -btnCancel : JButton -btnCreDisplay : JButton -btnCreClear : JButton -btnCreGoBank: JButton
+<<constructor>>BankGUI +actionPerformed(ActionEvent e) : Void +main([]:String agrs) : Void Static

3. Pseudo code

Pseudo code is an artificial and informal language that helps programmers develop algorithms. Pseudo code is not an actual programming language. Pseudo code is written in plane language to make program understandable and. It generally set the layout of the program before programming. This is done to understand the data flow for the final program. It is also the principle key for an algorithm. No any arithmetic symbol is used in it. Description and functionality are gathered first and only the statement is written (neapolitan, 2003).

IMPORT all from Swing

IMPORT font from AWT

IMPORT Colour from AWT

IMPORT actionEvent of Event from AWT

IMPORT ActionListener of Event from AWT

IMPORT ArrayList from Util package

IMPORT Grid Layout from AWT

CREATING a Class name BANKGUI

DO

CREATING a Constructor BANKGUI

DO

DECLEAR a generic ArrayList of BankCard

DECLEAR frame fBankCard, fDebitCard, fCreditCard and fDisplay

DECLEAR Panel pBankCard, pDebitCard, pCreditCard and pDisplay for fBankCard, fDebitCard, fCreditCard and fDisplay frame

DECLEAR JLabel for Card ID, Bank Account, Client Name, Issuer Bank, Balance Amount, pin Number and Confirm Pin where Necessary

DECLEAR JTextField for Card ID, Bank Account, Client Name, Issuer Bank, Balance Amount, Pin Number and Confirm Pin along with its Jlabel

DECLEAR JButtons for Create, Display, Go to DebitCard and Go to CreditCard

DECLEAR JLabel for Card ID, Bank Account, Client Name, Issuer Bank, Balance Amount, pin Number, withDraw amount and Date of WithDrawal for Debit Card.

DECLEAR JTextField for Corresponding CardID, BankAccount, Client Name, Issuer Bank, Balance Amount, Pin Number, Withdrawal Amount for Debit Card.

DECLEAR combo box for Date of Withdrawal containing Date, Month, and year for DebitCard.

DECLEAR JButton for go, Withdraw, Display, Clear, Go to BankCard for DebitCard

DECLEAR JLabel for Card ID, Bank Account, Client Name, Issuer Bank, Balance Amount, pin Number, CVC Number, Interest rate, Credit Limit, Grace Period, Expiration Date for fCreditCard

DECLEAR JTextField for Corresponding CardID, BankAccount, Client Name, Issuer Bank, Balance Amount, Pin Number, CVC Number, Interest rate, Credit Limit, Grace Period for fCreditCard.

DECLEAR combo box for Expiration Date containing Date, Month, and year for fCreditCard.

DECLEAR JButton ADDCreditCard, SET Credit Limit, Cancel Credit, Clear, Go to BankCard for fCreditCard

INITILIZE components for the Bank Card

SET Background color for panels

SET color for all JLabels

SET color for foreground of TextField

SET Font Size of TextField.

SET Bounds for all JLabel, JTextField, JComboBox and JButtons.

ADD all the JLabels, JtextField, JComboBox and JButton to the Panel

SET Size of panel.

SET Visible true for frame and panel used

SET layout of panel to null

SET location Related to null for frame

SET Resizable false to Frame

SET default close operation to exit when close

INITILIZE the component for the DebitCard

DECLEAR JLabel for title of DebitCard

ADD all the JLabels, JTextField, JButtons and JComboBox in panel

SET bounds for all JLabels, JTextField and JButtons

SET size of the pDebitCard

SET visible False for fDebitCard

SET Layout Null for FDebitCard

SET location Related to null for frame

SET Resizable false to Frame

SET default close operation to exit when close

INITILIZE the components for the fCreditCard

DECLEAR JLabel for the Title of CreditCard

ADD all the JLabels, JTextField, JButtons and JComboBox in panel

SET Bounds for All JLabels, JTextField, JComboBox and JButtons of CreditCard.

SET size of the pCreditCard

SET visible False for fCreditCard

SET Layout Null for fCreditCard

SET location Related to null for frame

SET Resizable false to Frame

SET default close operation to exit when close

DECLEAR TextArea for the Display pDisplay

DECLEAR ScrollPane for Verticle Scrollbar if Needed

SET Grid Layout of the panel

SET Size of panel of width 400 and Height 500

SET Visible False to pDisplay

SET Resizable false to Frame

SET default close operation to dispose when close

ADD actionListener to JBUtton btnGODebit of fBankCard

DO

ADD action Event for btnGODebit

DO

SET Visible True for fDebitCard

SET Visible True for pDebitCard

SET Visible False for fBankCard

SET Visible False for pBankCard

END DO

END DO

ADD actionListener to JBUtton btnGOBank of fDebitCard

DO

ADD action Event for btnGOBank of fDebitCard

DO

SET Visible False for fDebitCard

SET Visible False for pDebitCard

SET Visible True for fBankCard

SET Visible True for pBankCard

END DO

END DO

ADD actionListener to JBUtton btnCreGOCredit of fCreditCard

DO

ADD action Event for btnCreGOBank of fCreditCard

DO

SET Visible True for fBankCard

SET Visible True for pBankCard

SET Visible False for fCreditCard

SET Visible False for pCreditCard

END DO

END DO

ADD actionListener to JBUtton btnGOCredit of fBankCard

DO

ADD action Event for btnGOCredit of fBankCard

DO

SET Visible False for fBankCard

SET Visible False for pBankCard

SET Visible True for fCreditCard

SET Visible True for pCreditCard

END DO

END DO

ADD actionLlistener to JButton btnClear of fBankCard

DO

ADD action Event for btnClear of fBankCard

DO

SET null to JtextField of Card ID

SET null to JtextField of Bank Account

SET null to JtextField of Client Name

SET null to JtextField of Issuer Bank

SET null to JtextField of Balance Amount

SET null to JtextField of Pin

SET null to JtextField of Confirm Pin

END DO

END DO

ADD actionLlistener to JButton btnDebClear of fDebitCard

DO

ADD action Event for btnDebClear of fDebitCard

DO

SET null to JtextField of Card ID

SET null to JtextField of Bank Account

SET null to JtextField of Client Name

SET null to JtextField of Issuer Bank

SET null to JtextField of Balance Amount

SET null to JtextField of Pin

SET null to JtextField of WithDraw Amount

END DO

END DO

ADD actionLlistener to JButton btnCreClear of fCreditCard

DO

ADD action Event for btnCreClear of fCreditCard

DO

SET null to JtextField of Card ID

SET null to JtextField of Bank Account

SET null to JtextField of Client Name

SET null to JtextField of Issuer Bank

SET null to JtextField of Balance Amount

SET null to JtextField of Pin

SET null to JtextField of Cvc Number

SET null to JtextField of Credit Limit

SET null to JtextField of Interest Rate

SET null to JtextField of Grace

END DO

END DO

ADD action Listener to Button btnCreate of fBankCard

DO

ADD actionEvent for the Listener

DO

INITILIZE Boolean check equals to true

FOR EACH elements of arraylist

IF element belongs to BankCard

DO

DownCaste the BankCard

IF Card ID Entered is same to Card ID in ArrayList

DO

Check equals to false

Break the loop

END DO

ELSE

DO

Check equals to True

END DO

END DO

SET TRY

DO

IF Check Is equals True

DO

IF Card id is less then zero and anything is

Empty

DO

Show dialog box with suitable message.

END DO

ELSE

DO

IF Pin Number is same as the Confirm pin Number

DO

CREATE an object of BankCard and DebitCard

ADD the object in Array list

Show suitable Message

END DO

END DO

ELSE IF Check equals to False

DO

Show suitable message

END DO

SET Exception Handling

DO

Show suitable Message

END DO

END DO

END DO

ADD new Action listener to the Button btnDisplay

DO

ADD Action Performed to the Listener

DO

SET Visible to frame fDisplay

ADD StringBuilder

INITILIZE variable for combo box of day, Month and Year

FOR EACH loop in Array List

DO

IF element instance of DebitCard

DO

INITILIZE down casting of Debit Card

ADD all the Components of Debit Card to String Builder

END DO

END DO

IF String Builder is Empty

DO

PRINT suitable Message

END DO

ELSE

DO

SET all the value of String builder to textArea

END DO

END DO

END DO

ADD new action Listener to button btnDebGO

DO

ADD new Action Performed to Listener

DO

INITILIZE Boolean check equals to true

SET try

DO

FOR EACH elements of arraylist

IF element belongs to DebitCard

DO

DownCaste the DebitCardCard

IF Card ID Entered is same to Card ID in ArrayList

DO

Check equals to true

Break the loop

END DO

ELSE

DO

Check equals to false

END DO

END DO

IF Boolean is true

DO

FOR EACH loop of ArrayList

DO

IF element is instance of DebitCard

DO

INITILIZE Down casting of DebtiCard

IF card id entered is same as card id in arraylist

DO

SET the Text Fields with corresponding values like set Card id, bank account, client name, issuer bank and amount

SET editable of those text field false

END DO

END DO

END DO

END DO

IF Boolean is false

DO

Show Suitable Message

END DO

END DO

SET Exception Handiling

DO

Show suitable message

END DO

END DO

ADD action Listener to button btnWithDraw

DO

ADD action Performed to action Listener

DO

SET try

DO

FOR EACH loop in Arraylist

DO

IF element is instance of DebitCard

DO

DECLARE variables for Text fields of withdrawal amount, pin and Date of Withdrawal

ADD Downcasting of DebitCard

IF card id entered is equal to the card id in Arraylist

DO

IF pin entered is equal to account pin

DO

IF withdrawal amount is less than 0

DO

Show suitable message

END DO

ELSE

DO

IF withdrawal amount is less than or equal to balance amount

DO

ASSIGN argument for method ebitCard

Show suitable message

END DO

ELSE

DO

Show suitable message

END DO

END DO

END DO

ELSE

DO

Show suitable message

END DO

END DO

END DO

END DO

END DO

SET exception handiling

DO

Show suitable message

END DO

END DO

END DO

ADD new action listener to button btnDebDisplay

DO

ADD action performed to listener

DO

SET Visible to frame fDisplay

ADD StringBuilder

FOR EACH loop in Array List

DO

IF element instance of DebitCard

DO

INITILIZE down casting of Debit Card

ADD all the Components of Debit Card and withdraw to String Builder

END DO

END DO

IF String Builder is Empty

DO

PRINT suitable Message

END DO

ELSE

DO

SET all the value of String builder to textArea

END DO

END DO

END DO

ADD action Listener to button btnCreGo

DO

ADD action Performed to the listener

DO

INITILIZE Boolean check equals to true

FOR EACH elements of arraylist

DO

IF element belongs to BankCard

DO

DownCaste the BankCard

IF Card ID Entered is same to Card ID in ArrayList

DO

Check equals to true

Break the loop

END DO

ELSE

DO

Check equals to false

END DO

END DO

END DO

IF Boolean is true

DO

FOR EACH loop in arraylist

DO

IF element instance of BankCard

DO

DECLEAR downcasting of BankCard

IF card id entered is equal to the card id in arraylist

DO

SET value of textfield with their corresponding values of in array list

SET editable of those textfield to false

END DO

END DO

END DO

END DO

ELSE

DO

Show suitable message

END DO

END DO

END DO

ADD action listener to the button btnCreAdd

DO

ADD action Performed to the listener

DO

SET try

DO

DECLARE variables to store values of all the textfield of Crditcard

ASSIGN argument to the constructor of CreditCard

ADD credit card to the Array list

Show suitable message

END DO

CATCH exceptions

DO

Show suitable message

END DO

END DO

END DO

ADD new action Listener to button btnSet

DO

ADD action Performed to listener

DO

FOR EACH loop in arraylits

DO

IF element instance of CreditCard

DO

ADD try

DO

ADD Downcastof CreditCard

SET variables of limit and grace period

IF card id entered is equal to card id in Arraylist

DO

IF isgranted is false

DO

IF limit is less than 2.5 times balance amount

DO

SET argument for SetCreditlimit method of CreditCard object

Show suitable message

END DO

ELSE

DO

Show suitable message

END DO

END DO

ELSE

DO

Show suitable message

END DO

END DO

END DO

SET catch

DO

Show suitable message

END DO

END DO

END DO

END DO

ADD action listener to button btnCancel

DO

ADD action performed to listener

DO

DECLARE input dialog

FOR EACH loop Arraylist

DO

IF element instance of CreditCard

DO

ADD downcasting of CreditCard

IF card id entered is same as card id in arraylist

DO

IF isgranted is true

DO

CALL cancel credit Method from creditcard

Show suitable message

END DO

ELSE

DO

Show suitable message

END DO

END DO

ELSE

DO

Show suitable message

END DO

END DO

END DO

END DO

END DO

ADD action listener to button btnCreDisplay

DO

ADD action performed to listener

DO

SET Visible to frame fDisplay

ADD StringBuilder

FOR EACH loop in Array List

DO

IF element instance of Credit Card

DO

INITILIZE down casting of Credit Card

ADD all the Components of Credit Card to String Builder

END DO

END DO

IF String Builder is Empty

DO

PRINT suitable Message

END DO

ELSE

DO

SET all the value of String builder to text Area

END DO

END DO

END DO

END DO

DECLARE a main method

DO

CALL BankGUI constructor

END DO

4. Method Description

Method is the block of code on program to do something. It can be called whenever we required it to run. Method name, parameter, return value type and the body is consist in the method. In method header there is a variable which is also called formal parameter. A parameter is like a placeholder. If you want to set value in method you can pass it through parameter. In method body it contains the collection of statement that do something whether it return value or set value. Methods can be called or revoke to execute. (Liang, 2013)

I. btnGoDebit.ActionPerformed : In this Method Button btnGoDebit has defined an action Listener to add the functionality of button, to executed method written in the action Performed. This action Performed method changed the Visibility of Components. When the Button is pressed in fBankCard, This will make fDebitCard and pDebitCard visible whereas fBankCard and pBankCard will be invisible.

II. btnGoBank.ActionPerformed : In this Method Button btnGoBank has defined an action Listener to add the functionality of button, to executed method written in the action Performed. This action Performed method changed the Visibility of Components. When the Button is Pressed in fDebitCard, This will make fBankCard and pBankCard visible whereas fDebitCard and pDebitCard will be invisible.

III. btnCreGoBank.ActionPerformed : In this Method Button btnCreGoBank has defined an action Listener to add the functionality of button, to executed method written in the action Performed. This action Performed method changed the Visibility of Components. When the Button is pressed in fCreditCard, This will make fBankCard and pBankCard visible whereas fCreditCard and pCreditCard will be invisible.

IV. btnGoCredit.ActionPerformed: In this Method Button btnGoCredit has defined an action Listener to add the functionality of button, to executed method written in the action Performed. This action Performed method changed the Visibility of Components. When the Button is pressed in

fBankCard, This will make fCreditCard and pCreditCard visible whereas fBankCard and pBankCard will be invisible.

V. btnClear.ActionPerformed: In this Method, when the button btnClear in fBankCard is pressed it will set Null to the every TextField of fBankCard or other word it will clear the Text Field of fBankCard.

VI. btnDebClear.ActionPerformed: In this Method, when the button btnDebClear in fDebitCard is pressed it will set Null to the every TextField of fDebitCard or other word it will clear the Text Field of fDebitCard.

VII. btnCreClear.ActionPerformed: In this Method, when the button btnCreClear in fCreditCard is pressed it will set Null to the every TextField of fCreditCard or other word it will clear text in the Text Field of fCreditCard.

VIII. btnCreate.ActionPerformed: The method action performed in the method Check whether the entered Card ID is already existed in arraylist or not and if exit it will show the suitable message that card id has already been used to the user and if not it will check every entered value is correct for its textfield or not if it find some faults it will display you a message otherwise, it help user to Create his/her new Account and store it in Arraylist.

IX. btnDisplay.ActionPerformed: This method will help user to show detail of their account like their name, card id, bank Account, Balance amount and pin Numbers within a new frame making it visible. If it doesn't found any detail from arraylist it will display a message saying no data is available otherwise, all the details of Client will be in screen.

X. btnDebGo.ActionPerformed: The method will check whether the card Id entered is available in object of DebitCard in arraylist or not. If card Id is found it will fill Client information in corresponding textfield and helps run further. Client will not able to manipulate his/her account detail there. If Card id is not found in Arraylist it will display you a suitable message using dialog box.

XI. btnWithDraw.ActionPerformed: The method action Performed in the Listener analyse whether the information belongs to DebitCard or not in arraylist. This will help user to withdraw amount from there account. Firstly it will check whether the Card ID is available in arraylist or not and onward it will examine the pin of the user and withdrawal amount if it passes all the criteria this method will call the method from debitCard withdraw and deducted the amount from client account. And if failed to cross criteria it will display suitable message for user.

XII. btnDebDisplay.ActionPerformed: This method will help user to show detail of their account like their name, card id, bank Account, Balance amount, pin Numbers, Withdrawal Amount, Date of Withdrawal and Is amount withdrawed or not within a new frame making it visible. If it doesn't found any detail from arraylist it will display a message saying no data is available otherwise, all the details of Client will be in screen.

XIII. btnCreGo.ActionPerformed: The method will check whether the card Id entered is available in object of CreditCard in arraylist or not. If card Id is found it will fill Client information in corresponding textfield and helps run further. Client will not able to manipulate his/her account detail there. If Card id is not found in Arraylist it will display you a suitable message using dialog box.

XIV. btnCreAdd.ActionPerformed: The method in the Action Listener will check the entered value is suitable or not. If everything is correct it will Create an object of Client containing all details of client, of CreditCard and store it in Arraylist. After onward it will show you a message for whether the object is able to add on arraylist or not. If Some data is not suitable for the Text fields will be instantly show you a Error message about you fault.

XV. btnSet.ActionPerformed: This method will check the entered Card Id in object of CreditCard present in arraylist and if card id is found it will allow you to move forward and help you set your credit. Before that it will also

analyse you credit amount must be only 2.5 times your current Balance Amount. It will call the method setCreditLimit of Credit Card and pass the argument Credit limit and grace period in it and add this method to client object. Else if your credit limit is more than the criteria than it will display client a message about you Credit Limit.

XVI. btnCancel.ActionPerformed: The method in this action listener will ask the user to enter his/her pin number to allow the cancel access. The method search the object of the client through their Card id and analysis whether the client has taken Credit or not if taken it will help client to cancel his/her credit by calling method cancel from Credit Card. After the criteria is fulfilled it will set Cvc Number, Grace Period and Credit Limit null in client object. Otherwise it will so a suitable message to user.

XVII. btnCreDisplay.ActionPerformed: This method will help Client to show detail of their account like their name, card id, bank Account, Balance amount, Cvc Number, Withdrawal Grace Period, Credit Limit, Interest Rate and Expiration Date within a new frame making it visible. If it doesn't found any detail from arraylist it will display a message saying no data is available otherwise, all the details of Client will be in screen.

XVIII. Main: It is the main method of the class. The main method here calls the Constructor of BankGUI.

5. Test

5.1 Test 1

Test NO :	1
Objective	To Compile and run the program using Command Prompt.
Action	• Open the command Prompt and set path to the file of program. • Compile Bank Card java file. • Compile Debit Card java file. • Compile Credit Card java file. • Compile BankGUI java file. • Run BankGUI java file.
Actual Result	All the Files were compiled and run properly.
Excepted Result	All the files must be Compiled and Run using command prompt.
Conclusion	The test is successful

Table 1 File Run and Compile in Comand Prompt

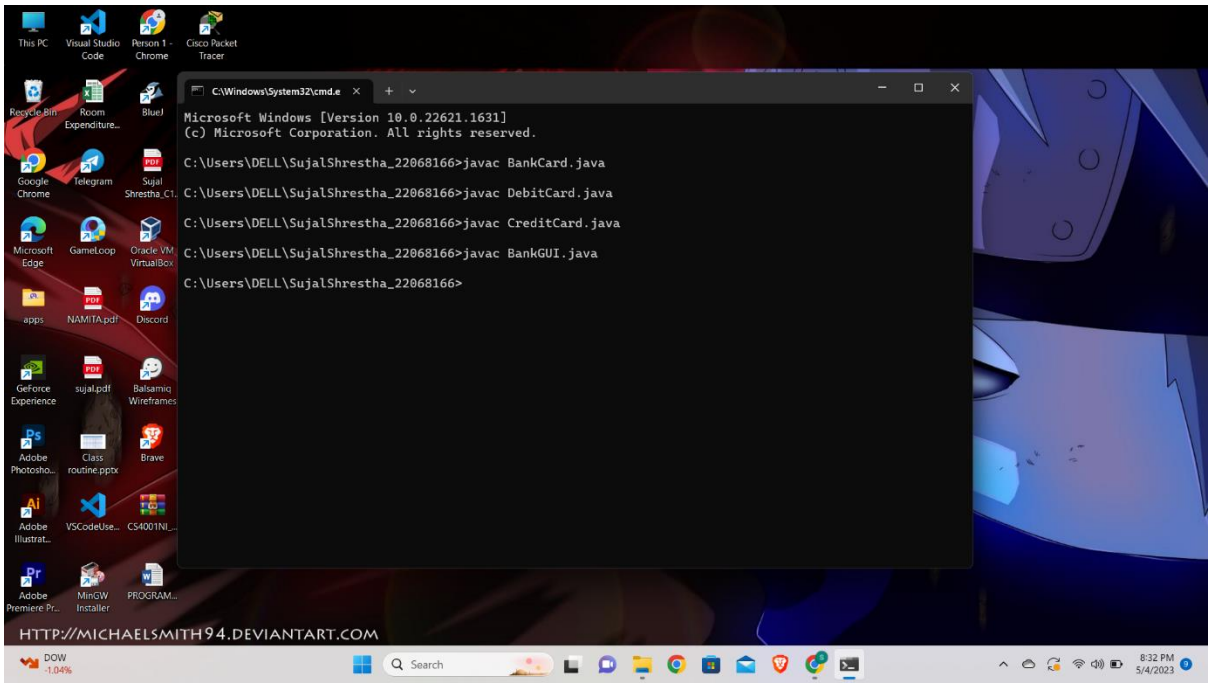


Figure 1 Compaling from Comand Prompt

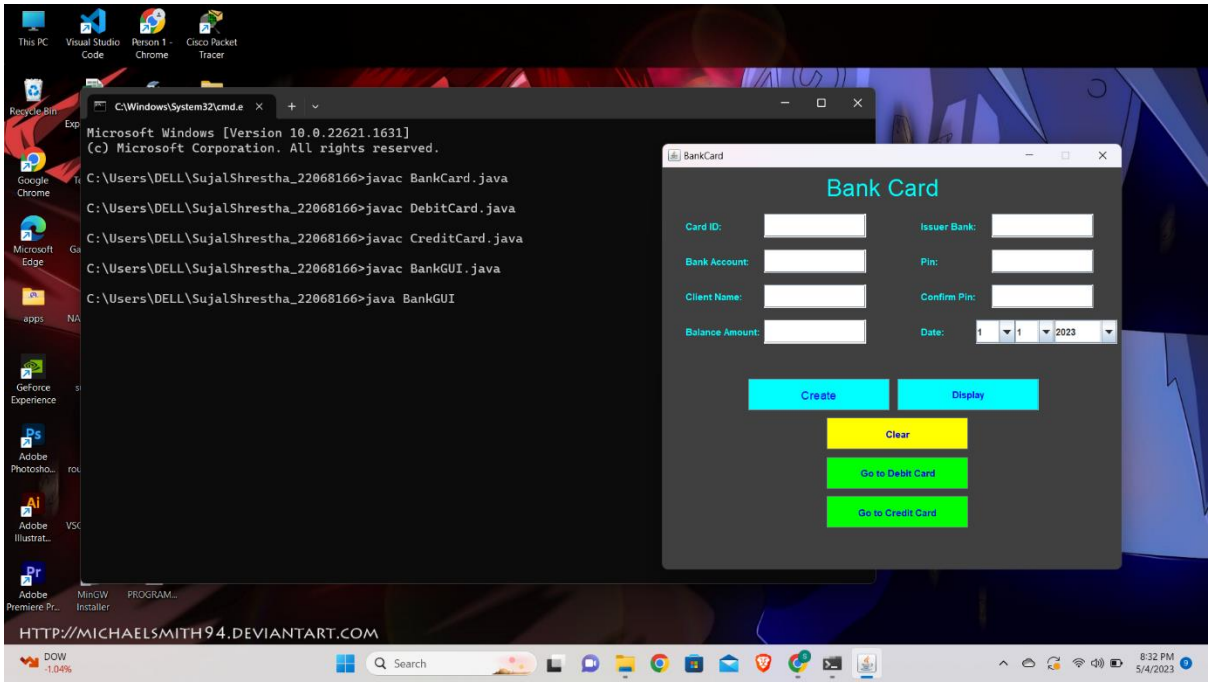


Figure 2 Run from Comand Prompt

5.2 Test 2

a) Add DebitCard.

Test NO :	2.a
Objective	To set Debit Card Object in Arraylist.
Action	• Run the File. • All the Data is filled properly. • Click Create Button on GUI.
Actual Result	All the data entered are stored as an object of DebitCard and add to the Array List.
Excepted Result	Object of Debit Card must be add in Array List.
Conclusion	The test is successful

Table 2 Set DebitCard

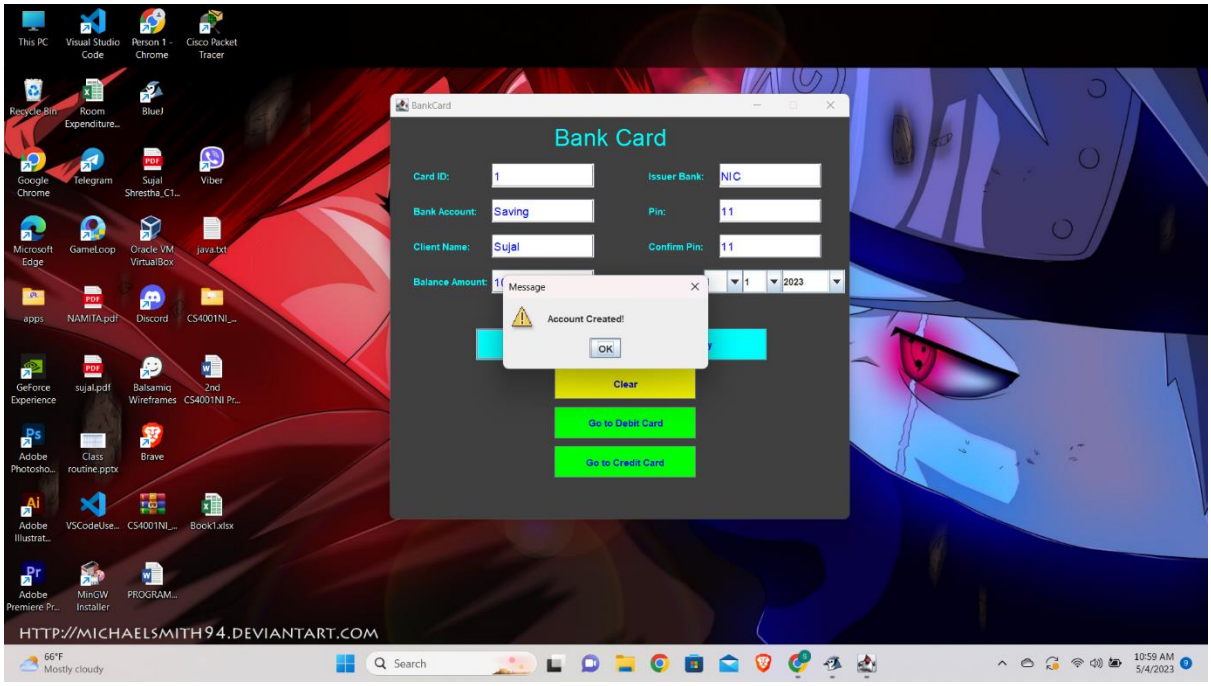


Figure 3 Adding DebitCard

b) Add Credit Card

Test NO :	2.b
Objective	To add object of Credit Card in Arraylist
Action	• Run the file. • Entered all the data necessary. • Press Add Button in Credit Card GUI.
Actual Result	All the data entered are stored as an object of Credit Card and add to the Array List.
Excepted Result	Object of Credit Card must be add in Array List.
Conclusion	The test is successful

Table 3 Add Credit Card

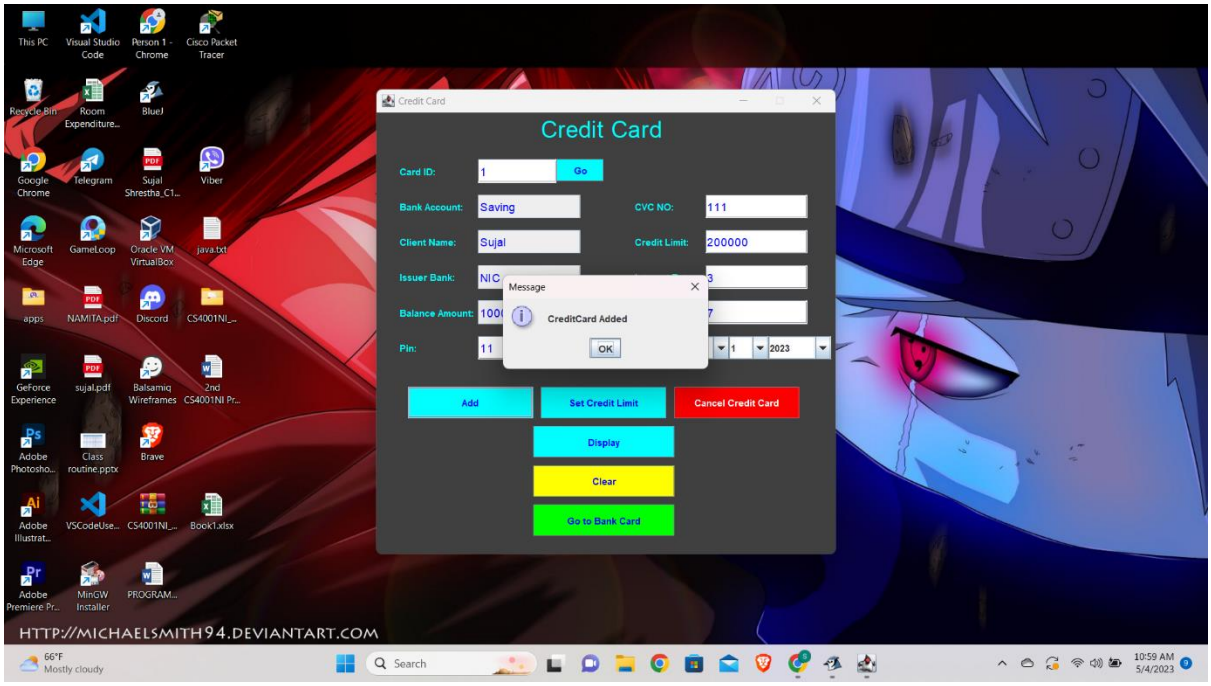


Figure 4 Adding CreditCard

c) WithDraw amount from DebitCard.

Test NO :	2.c
Objective	To withdraw amount from the client object of Debit Card.
Action	• Run the file. • Entered all the data necessary. • Press Withdraw Button in Debit Card GUI.
Actual Result	Withdrawal amount is reduce from the balance amount of client.
Excepted Result	Withdrawal Amount must be deduce from balance amount of client object.
Conclusion	The test is successful

Table 4 Withdraw amount from Debit Card

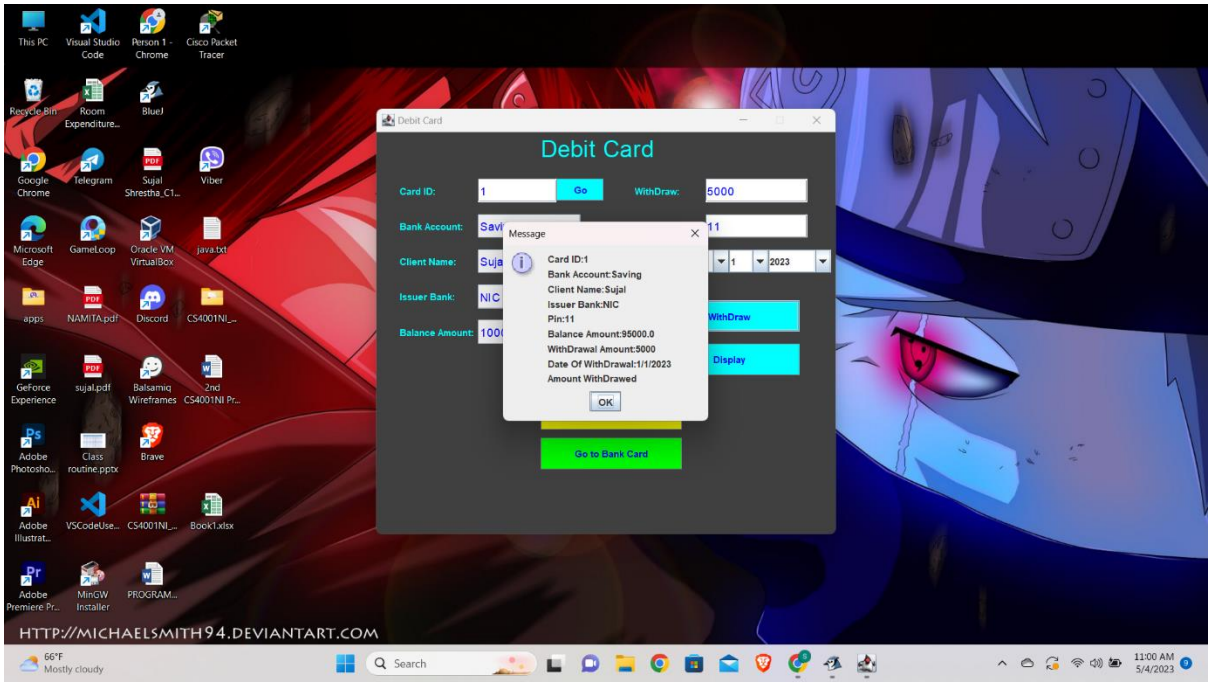


Figure 5 Withdrawing Amount from DebitCard

d) Set Credit Limit

Test NO :	2.d
Objective	To Set credit limit in object of Credit Card in Arraylist
Action	• Run the file. • Entered all the data necessary. • Press Set Credit Button in Credit Card GUI.
Actual Result	All the data entered for the setting credit limit are stored in an object of Credit Card in Array List.
Excepted Result	Credit limit, Grace Period must be set to the object of client in array list.
Conclusion	The test is successful

Table 5 Set Credit Limit in Credit Card

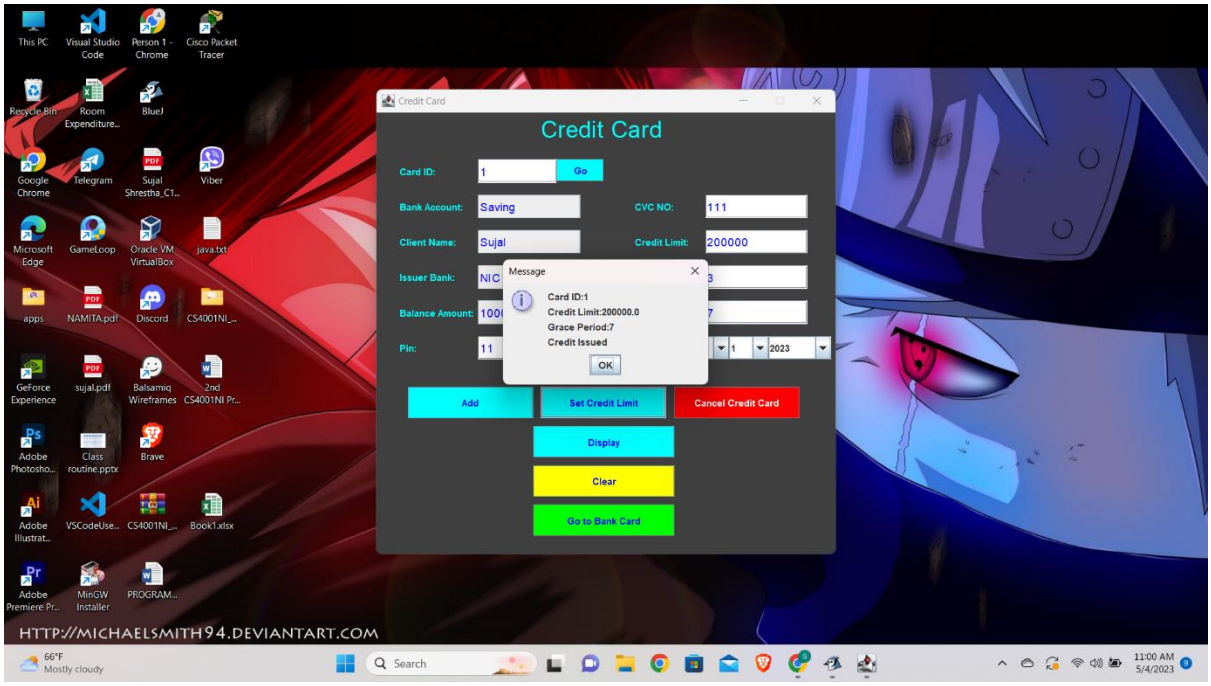


Figure 6 Set Credit Limit

e) Remove CreditCard.

Test NO :	2.e
Objective	To remove Credit limit of client of Credit Card in Arraylist
Action	• Run the file. • Entered all the data necessary. • Press Cancel Credit Button in Credit Card GUI.
Actual Result	All the data entered are removed from an object of Credit Card.
Excepted Result	Credit limit, Grace Period must be Remove from the object of client in array list.
Conclusion	The test is successful

Table 6 Cancel Credit

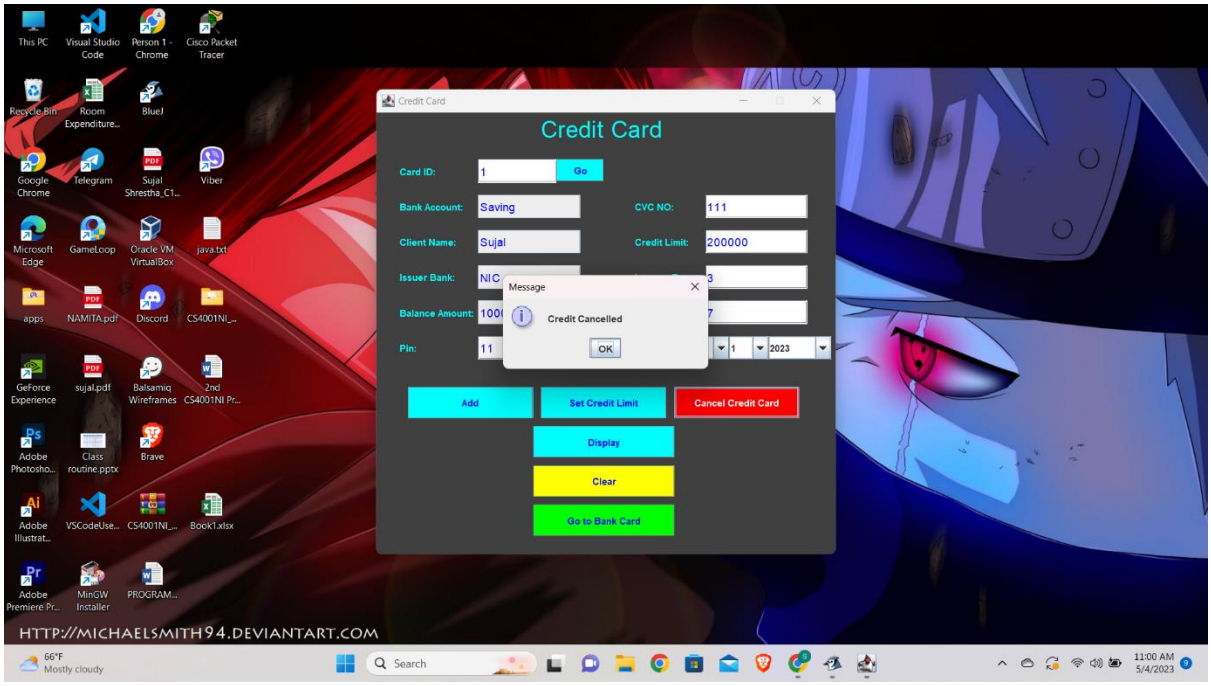


Figure 7 CreditCard Cancel

5.3 Test 3

Appropriate dialog box appears when unsuitable data is entered.

Test NO :	3
Objective	To see appropriate dialog box for unsuitable data entered in GUI.
Action	• Run the file. • Entered all the unnecessary data in Text field. • Press Buttons in GUI.
Actual Result	Appropriate Dialog box was shown on unnecessary input.
Excepted Result	Dialog box must be display on invalid/unnecessary inputs.
Conclusion	The test is successful

Table 7 Unsuitable input

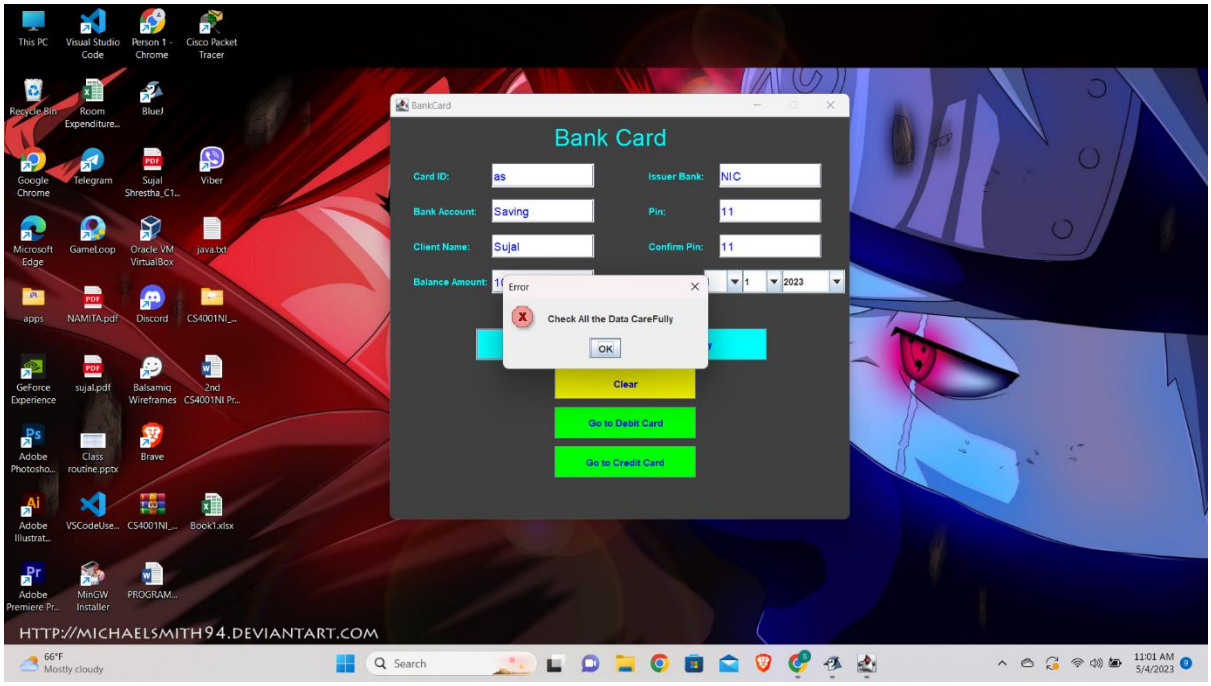


Figure 8 Unsuitable message for CardId of BankCard

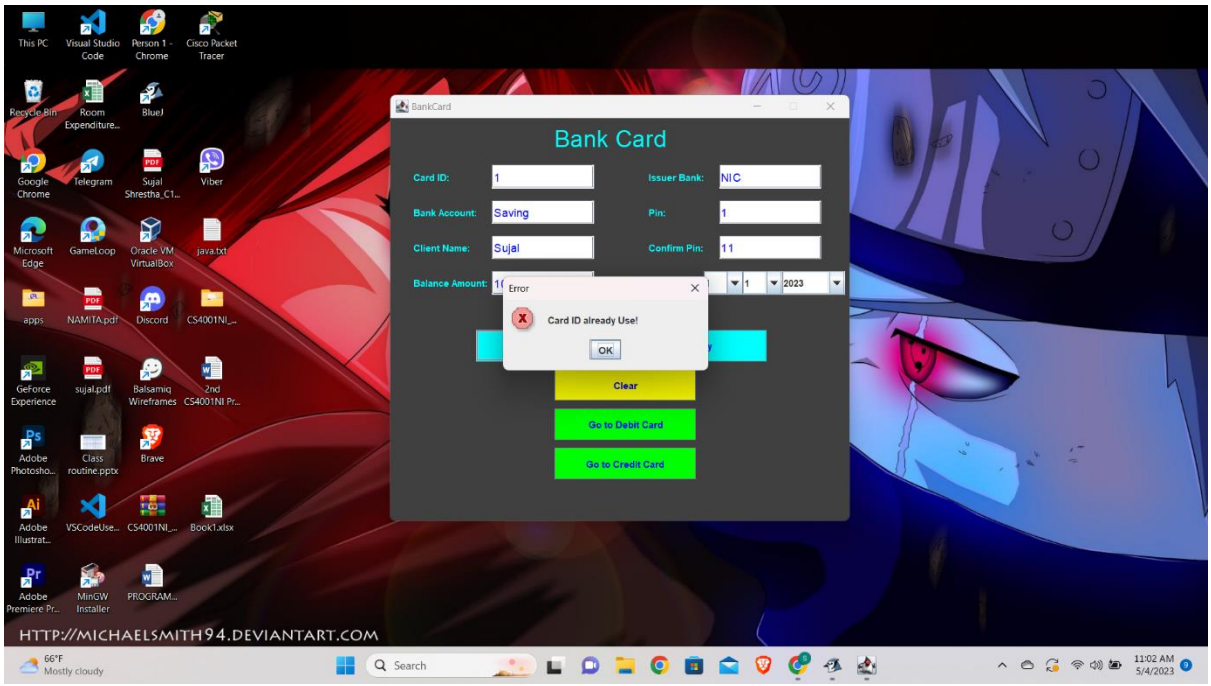


Figure 9 CardID already been used

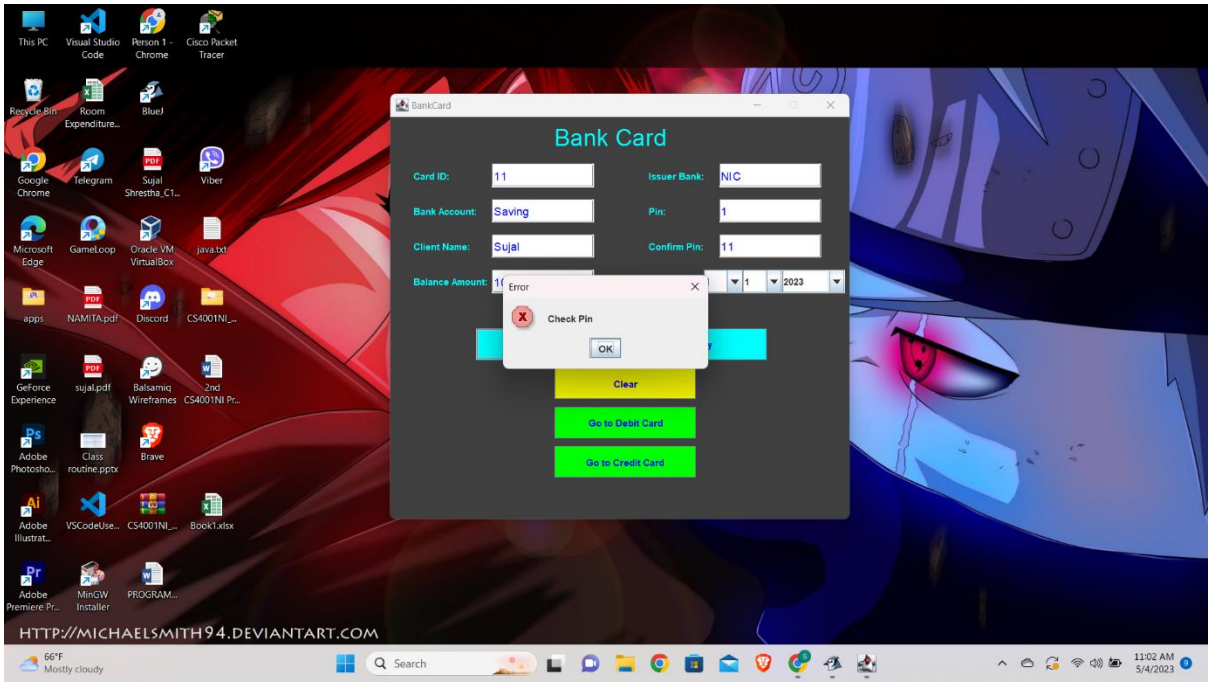


Figure 10 Incorrect Pin Number

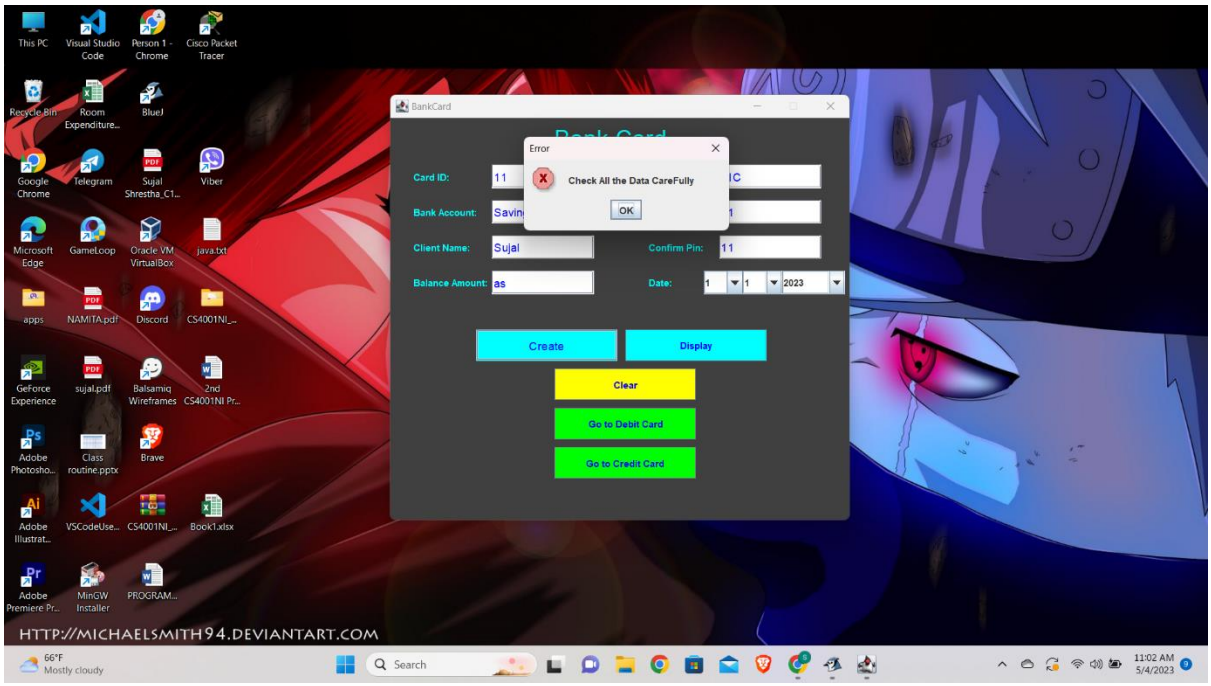


Figure 11 Unsuitable Input for Balance Amount

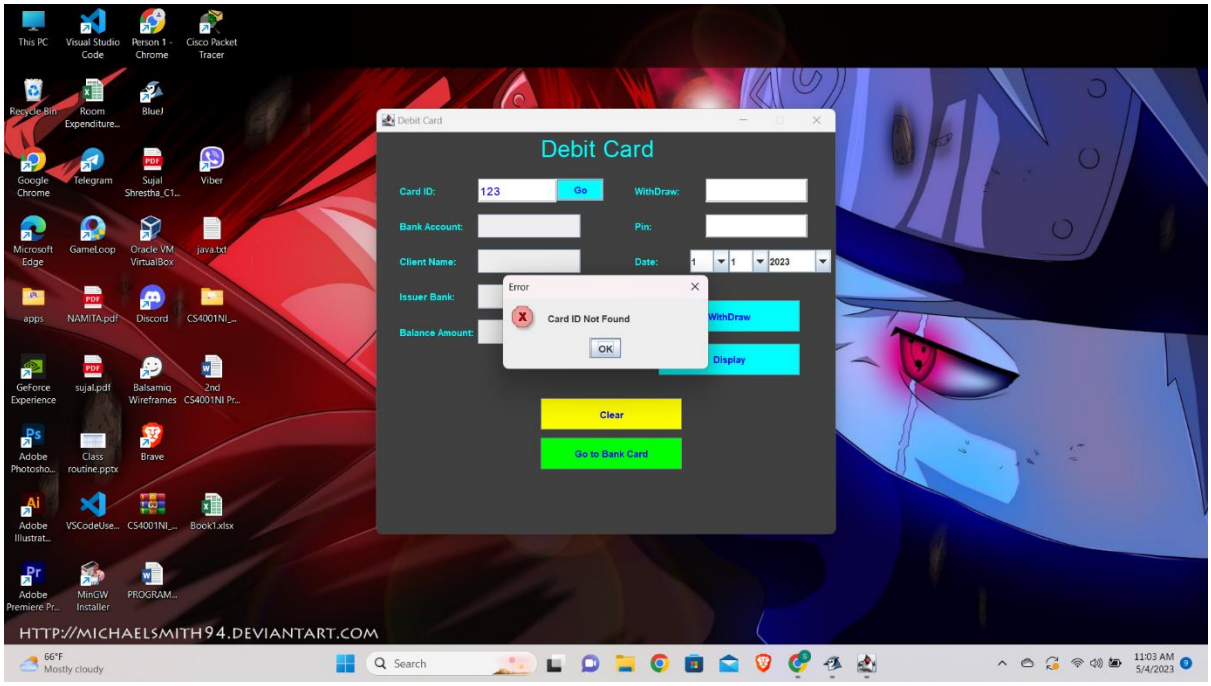


Figure 12 Input Incorrect Card ID for DebitCard

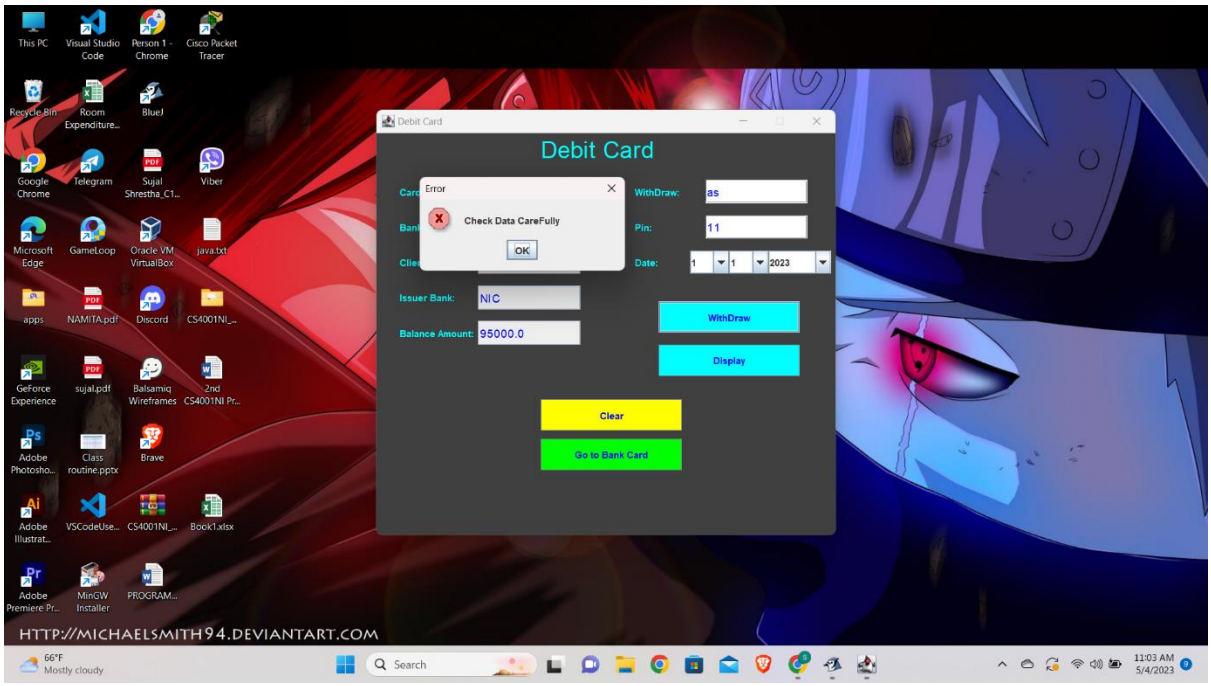


Figure 13 Unsuitable Input for WIthDraw amount

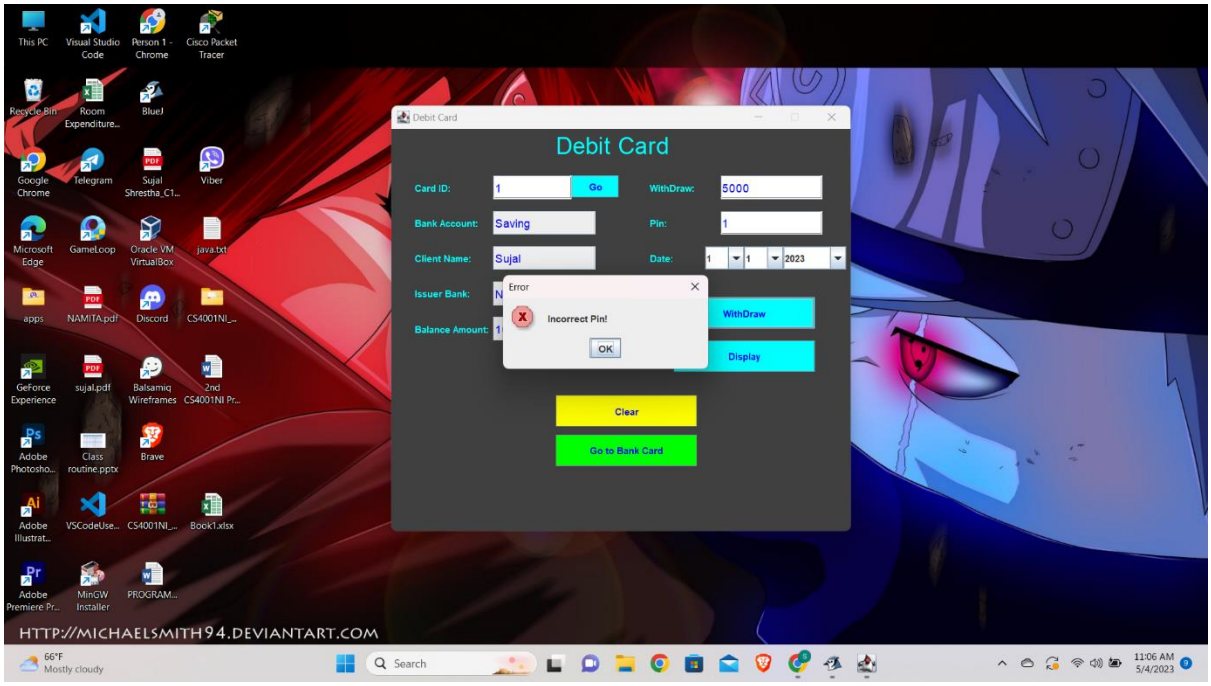


Figure 14 Entered Incorrect Pin while WIthdrawing

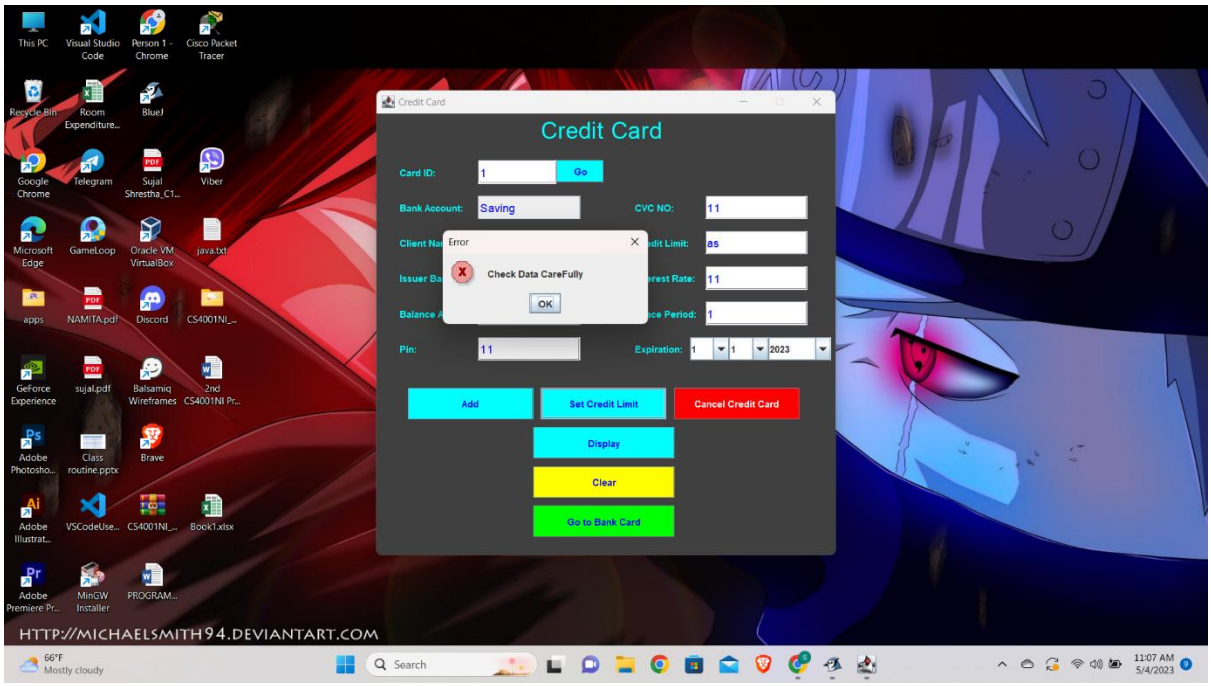


Figure 15 Entered unsuitable input for Credit Limit while setting Credit

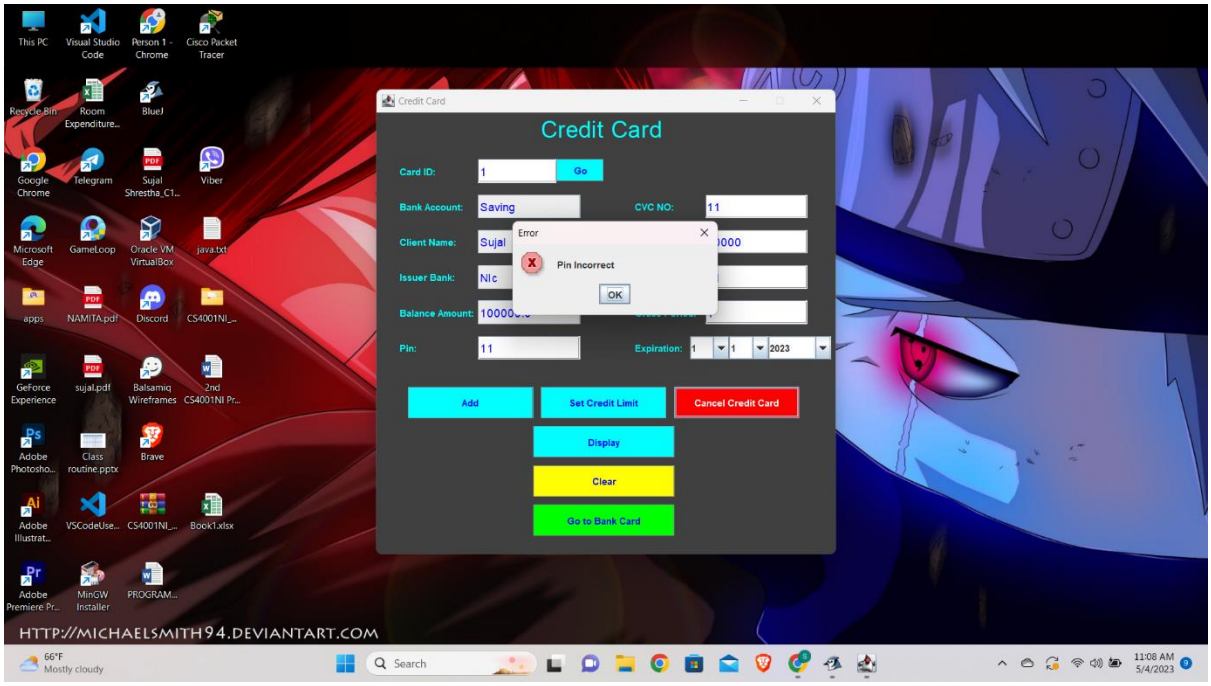


Figure 16 Incorrect pin for cancelling CreditCard

6. Errors

The faults that occurred in the programming that make program faulty is called error. There are 3 types of error in the programming. They are commonly called bugs in programming. The process of removing the error from programming is called debug or debugging. The error in the programming can be of different type some of them can be seen by the compilers but some of them can't be caught (Ben-Ari, 2007) .Such as:

- Syntax Error
- Semantic Error
- Logical Error

6.1 Syntax Error

The error in the program due to the misspelling of the codes is commonly known as Syntax error. This is caused because of the violation of programming language rules. They also known as compile-time errors. This error is catch by the compiler so it is the easiest error for handling. Examples of syntax errors include forgetting to use a semicolon at the end of a line, using the wrong syntax for a loop or conditional statement, misspelling a keyword or variable name, or using incorrect indentation.

Error No	1
Error Type	Syntax Error
Actual result	Writing a frame name that is declare above in instance but suddently wrote fBankCar.
Expected result	As declare in instance the name of frame is fBankCard.
Conclusion	Solved

Table 8 Syntax Error Occurred

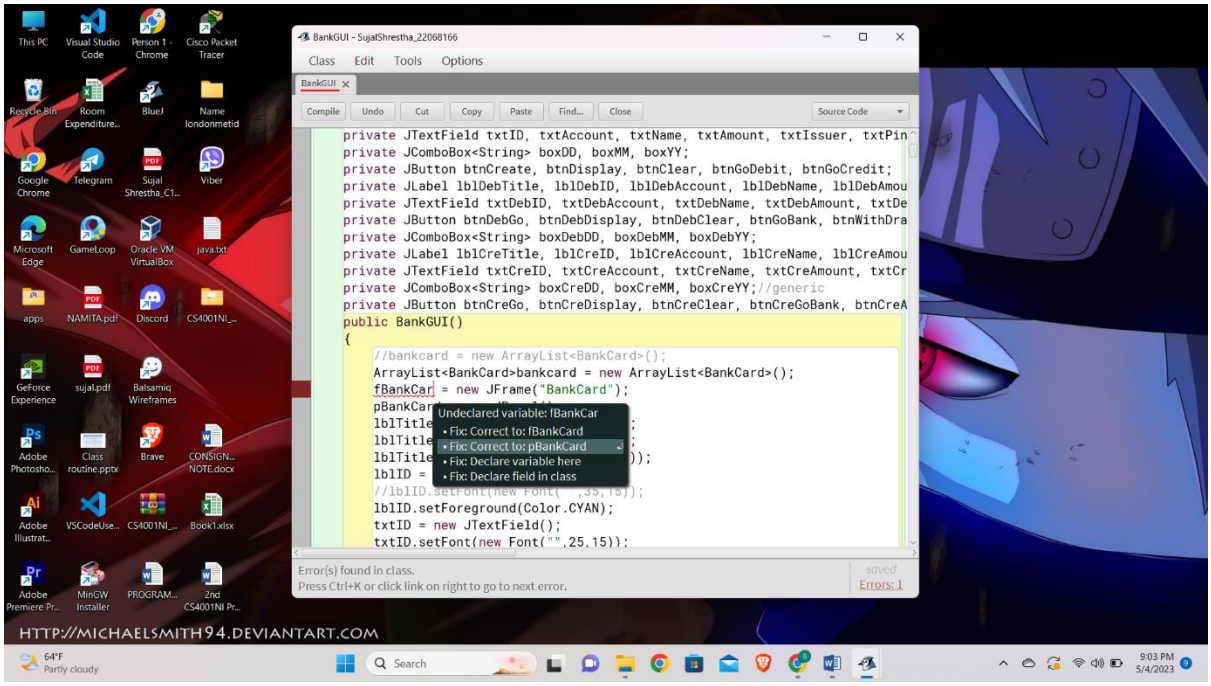
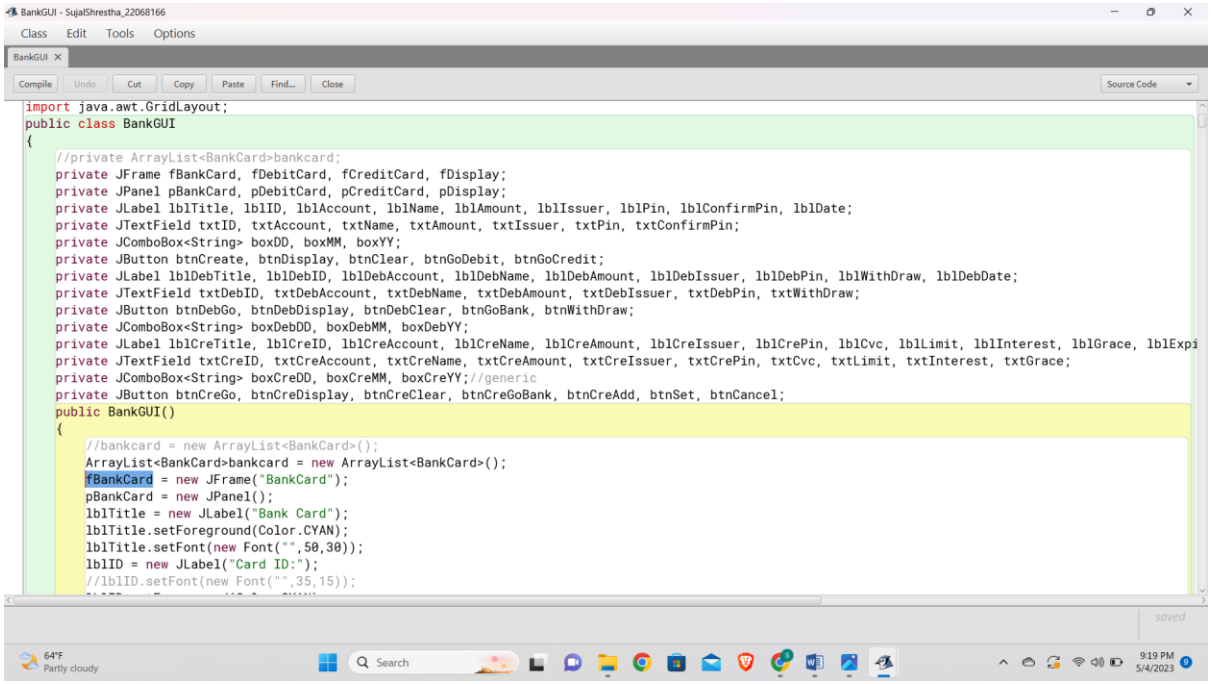


Figure 17 Syntax Error occurred




```
import java.awt.GridLayout;
public class BankGUI
{
    //private ArrayList<BankCard>bankcard;
    private JFrame fBankCard, fDebitCard, fCreditCard, fDisplay;
    private JPanel pBankCard, pDebitCard, pCreditCard, pDisplay;
    private JLabel lblTitle, lblID, lblAccount, lblName, lblAmount, lblIssuer, lblPin, lblConfirmPin, lblDate;
    private JTextField txtID, txtAccount, txtName, txtAmount, txtIssuer, txtPin, txtConfirmPin;
    private JComboBox<String> boxDD, boxMM, boxYY;
    private JButton btnCreate, btnDisplay, btnClear, btnGoDebit, btnGoCredit;
    private JLabel lblDebTitle, lblDebID, lblDebAccount, lblDebName, lblDebAmount, lblDebIssuer, lblDebPin, lblWithDraw, lblDebDate;
    private JTextField txtDebID, txtDebAccount, txtDebName, txtDebAmount, txtDebIssuer, txtDebPin, txtWithDraw;
    private JButton btnDebGo, btnDebDisplay, btnDebClear, btnGoBank, btnWithDraw;
    private JComboBox<String> boxDebDD, boxDebMM, boxDebYY;
    private JLabel lblCreTitle, lblCreID, lblCreAccount, lblCreName, lblCreAmount, lblCreIssuer, lblCrePin, lblCvc, lblLimit, lblInterest, lblGrace, lblExpi
    private JTextField txtCreID, txtCreAccount, txtCreName, txtCreAmount, txtCreIssuer, txtCrePin, txtCvc, txtLimit, txtInterest, txtGrace;
    private JComboBox<String> boxCreDD, boxCreMM, boxCreYY;//generic
    private JButton btnCreGo, btnCreDisplay, btnCreClear, btnCreGoBank, btnCreAdd, btnSet, btnCancel;
    public BankGUI()
    {
        //bankcard = new ArrayList<BankCard>();
        ArrayList<BankCard>bankcard = new ArrayList<BankCard>();
        fBankCard = new JFrame("BankCard");
        pBankCard = new JPanel();
        lblTitle = new JLabel("Bank Card");
        lblTitle.setForeground(Color.CYAN);
        lblTitle.setFont(new Font("",50,30));
        lblID = new JLabel("Card ID:");
        //lblID.setFont(new Font("",35,15));
```

Figure 18 Solving Syntax Error

6.2 Semantic Error

The error in the program that result incorrect but terminate well is called Semantic error. Such kind of error is very difficult to find while compiling. Examples of semantic errors include using the wrong formula in a calculation, using the wrong operator in a conditional statement, or using the wrong data type for a variable. It is more difficult to find than syntax error. Semantic error I got in my project.

Error No	2
Error Type	Semantic Error
Actual result	Assigning argument for the method withdraw of Debit Card. But the order of argument is misplaced for withdrawal amount and Pin Number.
Expected result	Placing correct order of argument in the parameter of withdraw method of Debit Card.
Conclusion	Solved

Table 9 Semantic Error Occurred

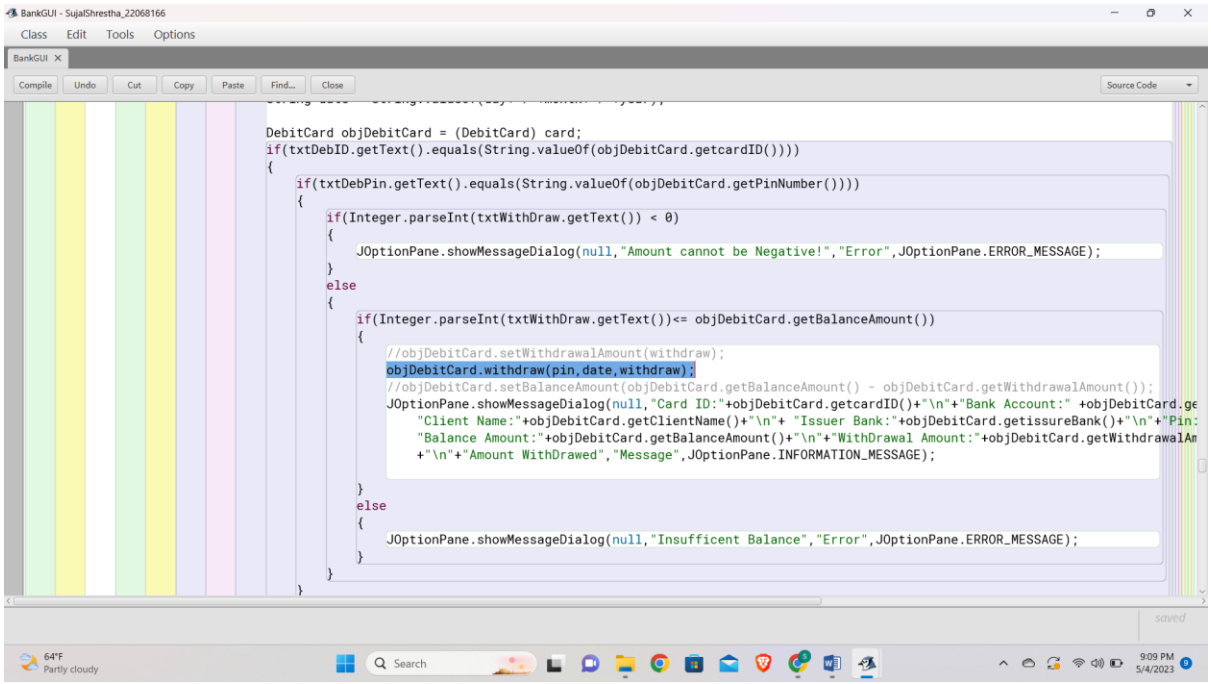


Figure 19 Semantic Error Occurred

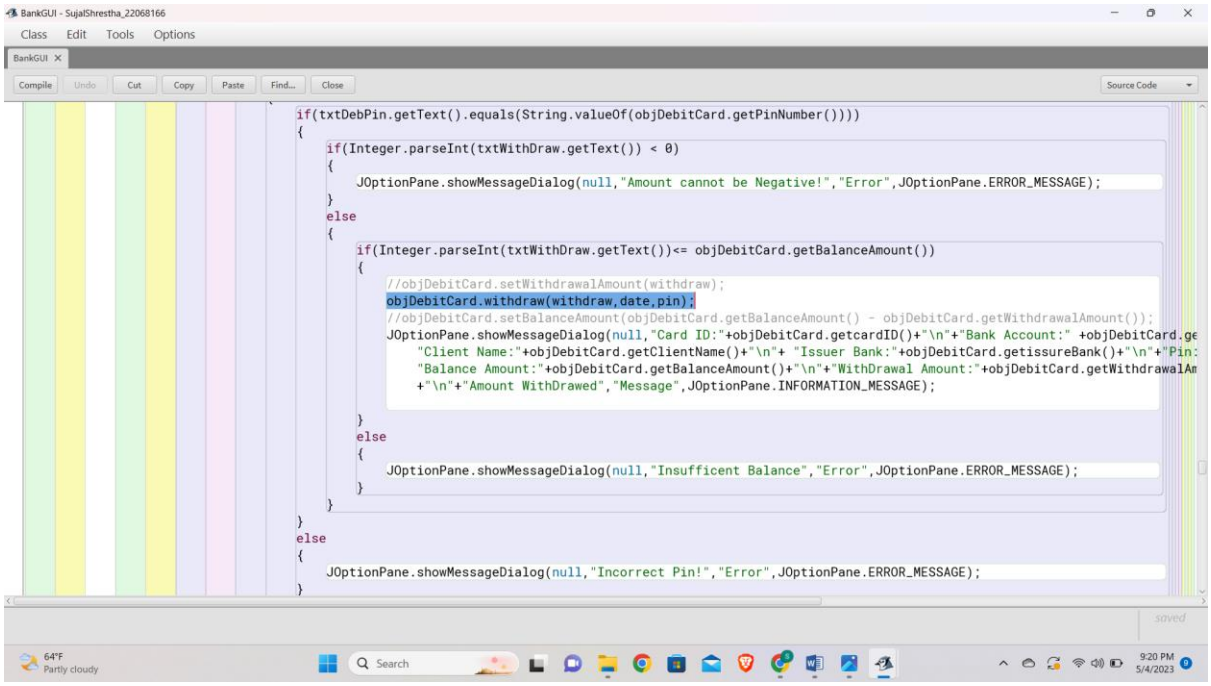


Figure 20 Solving Semantic Error

6.3 Logical Error

The error in the program due to wrong logic and results abnormally but terminate well is called logical error. Examples of logical errors include using the wrong algorithm to solve a problem, using incorrect or inconsistent data, using incorrect assumptions, or overlooking certain cases or scenarios. Such kind of error is due to the faulty logic of the programmer. This type of error is hard to find.

Logical Error I got in my project.

Error No	3
Error Type	Logical Error
Actual result	Adding Condition for the withdrawing where withdrawal Amount must be smaller or equals to client Balance Amount but incorrectly Condition was withdrawal amount must be greater or equal to client Balance Amount.
Expected result	The condition should be withdrawal amount must be smaller or equal to client Balance Amount.
Conclusion	Solved

Table 10 Logical Error Occurred

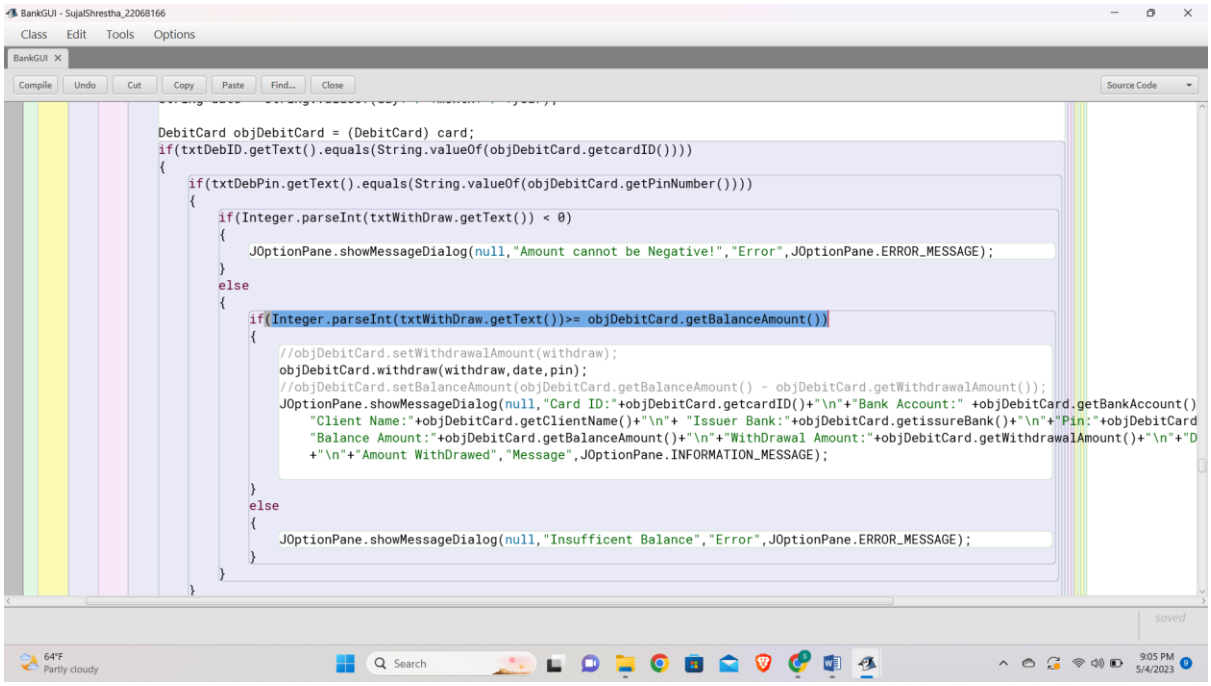


Figure 21 Logical Error Occurred

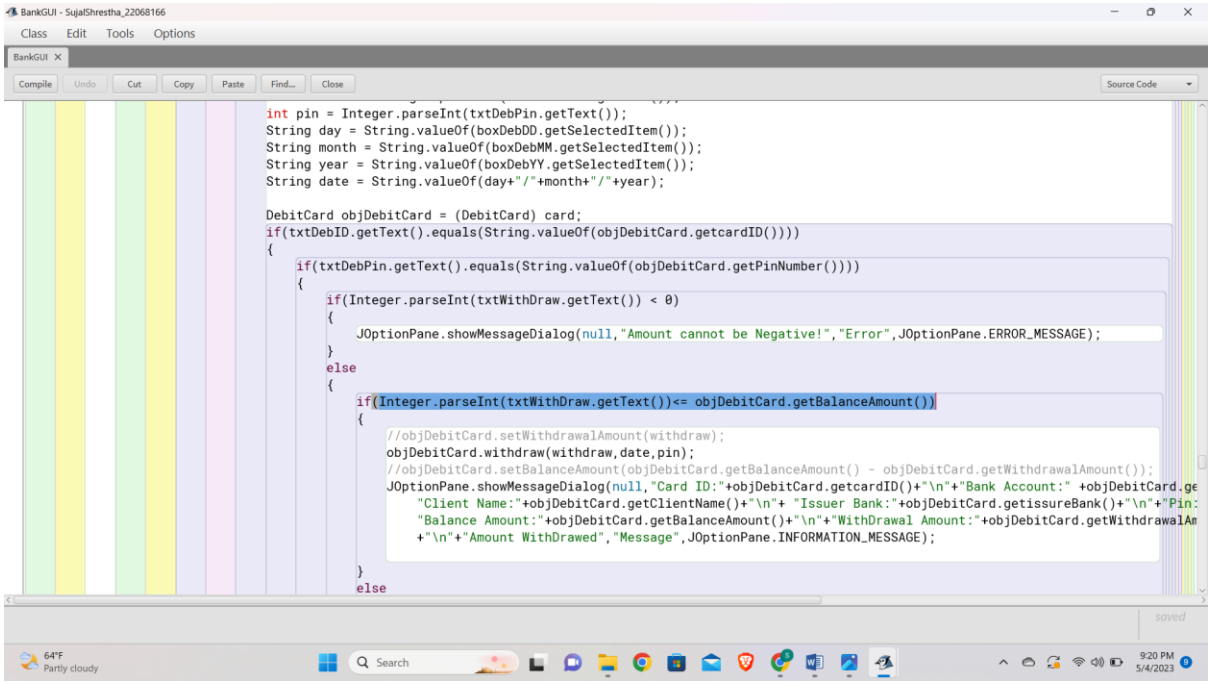


Figure 22 Logical Error Solve

7. References

Ben-Ari, M. (., 2007. *error in java,* isreal: Department of Science Teaching.

keller, S. B. a. R., 2001. *Class diagram,* s.l.: Program Comprehension.

Liang, Y. d., 2013. *Method,* new york, sanfranciso: PEARSON.

neapolitan, R. E., 2003. *Pseudocode,* London: Jones and bartlett.

8. Conclusion

In conclusion, Graphical User Interface I created provides user-friendly platform for the user to deal with their accounts. The design and functionality of the GUI will provide all the features what user expect for. The GUI made user easy to check the account detail, balance amount, withdraw amount from the account, get credit and also cancel the credit, the also can get receipt of the transactions. If any errors occur it provide suitable or relatable message to user.

By all counts and research I have learn one thing very special about GUI is that it is created because text command line interface is very complicated and difficult to learn. So all people cannot able to use the tools. To make Technology user-friendly and easy for all GUI is invented. GUI is very essential and the most important part in technology sector. GUI is that interface where we interact with the machines. All the frames, icons, buttons, whatever we see and use is a GUI.

During this project, I got difficulties like compilation errors using command prompt, unchecked error, writing in pseudo code, logical errors in my programs and many tiny problems. Sometimes errors like Logical error had got my whole day. The main and unnecessary problem in my project is the application is used for coding bluej. This application got hanged many times and many times I lost my data. Also more, this application cannot load my program for long time.

I am very thank full to every helping hand for me in this project. Whenever I fall in my problems I got hand of my tutors, lecturer. I research about my problems in Google and other sites and solve it. If not I ask help from my teachers who helps me a lot in my project they explain every problem of mine and help me to get through it. At last I want to say It is very interesting to do and learn about this project.

9. Appendix

```
import javax.swing.*;

import java.awt.Font;

import java.awt.Color;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.ArrayList;

import java.awt.GridLayout;

public class BankGUI

{

    private ArrayList<BankCard>bankcard;

    private JFrame fBankCard, fDebitCard, fCreditCard, fDisplay;

    private JPanel pBankCard, pDebitCard, pCreditCard, pDisplay;

    private JLabel lblTitle, lblID, lblAccount, lblName, lblAmount, lblIssuer, lblPin, lblConfirmPin, lblDate;

    private JTextField txtID, txtAccount, txtName, txtAmount, txtIssuer, txtPin, txtConfirmPin;

    private JComboBox<String> boxDD, boxMM, boxYY;

    private JButton btnCreate, btnDisplay, btnClear, btnGoDebit, btnGoCredit;

    private JLabel lblDebTitle, lblDebID, lblDebAccount, lblDebName, lblDebAmount, lblDebIssuer, lblDebPin, lblWithDraw, lblDebDate;

    private JTextField txtDebID, txtDebAccount, txtDebName, txtDebAmount, txtDebIssuer, txtDebPin, txtWithDraw;

    private JButton btnDebGo, btnDebDisplay, btnDebClear, btnGoBank, btnWithDraw;

    private JComboBox<String> boxDebDD, boxDebMM, boxDebYY;
```

```
    private JLabel lblCreTitle, lblCreID, lblCreAccount, lblCreName, lblCreAmount, lblCreIssuer, lblCrePin, lblCvc, lblLimit, lblInterest, lblGrace, lblExpiration;

    private JTextField txtCreID, txtCreAccount, txtCreName, txtCreAmount, txtCreIssuer, txtCrePin, txtCvc, txtLimit, txtInterest, txtGrace;

    private JComboBox<String> boxCreDD, boxCreMM, boxCreYY;//generic

    private JButton btnCreGo, btnCreDisplay, btnCreClear, btnCreGoBank, btnCreAdd, btnSet, btnCancel;

    public BankGUI()

    {

        ArrayList<BankCard>bankcard = new ArrayList<BankCard>();

        fBankCard = new JFrame("BankCard");

        pBankCard = new JPanel();

        lblTitle = new JLabel("Bank Card");

        lblTitle.setForeground(Color.CYAN);

        lblTitle.setFont(new Font("",50,30));

        lblID = new JLabel("Card ID:");

        //lblID.setFont(new Font("",35,15));

        lblID.setForeground(Color.CYAN);

        txtID = new JTextField();

        txtID.setFont(new Font("",25,15));

        txtID.setForeground(Color.BLUE);

        lblAccount = new JLabel("Bank Account:");

        lblAccount.setForeground(Color.CYAN);

        txtAccount = new JTextField();

        txtAccount.setFont(new Font("",25,15));
```

```
txtAccount.setForeground(Color.BLUE);
lblName = new JLabel("Client Name:");
lblName.setForeground(Color.CYAN);
txtName = new JTextField();
txtName.setFont(new Font("",25,15));
txtName.setForeground(Color.BLUE);
lblAmount = new JLabel("Balance Amount:");
lblAmount.setForeground(Color.CYAN);
txtAmount = new JTextField();
txtAmount.setFont(new Font("",25,15));
txtAmount.setForeground(Color.BLUE);
lblIssuer = new JLabel("Issuer Bank:");
lblIssuer.setForeground(Color.CYAN);
txtIssuer = new JTextField();
txtIssuer.setFont(new Font("",25,15));
txtIssuer.setForeground(Color.BLUE);
lblPin = new JLabel("Pin:");
lblPin.setForeground(Color.CYAN);
txtPin = new JTextField();
txtPin.setFont(new Font("",25,15));
txtPin.setForeground(Color.BLUE);
lblConfirmPin = new JLabel("Confirm Pin:");
lblConfirmPin.setForeground(Color.CYAN);
txtConfirmPin = new JTextField();
```

```
txtConfirmPin.setFont(new Font("",25,15));
txtConfirmPin.setForeground(Color.BLUE);
lblDate = new JLabel("Date:");
lblDate.setForeground(Color.CYAN);
String day[] = {"1","2","3","4","5","6","7","8","9","10","11","12","13","14","15"};
boxDD = new JComboBox<String>(day);
boxDD.setForeground(Color.BLUE);
boxDD.setEditable(true);
String month[] = {"1","2","3","4","5","6","7","8","9","10","11","12"};
boxMM = new JComboBox<String>(month);
boxMM.setEditable(true);
String year[] = {"2023","2024","2025","2026"};
boxYY = new JComboBox<String>(year);
boxYY.setEditable(true);
btnCreate = new JButton("Create");
btnCreate.setForeground(Color.BLUE);
btnCreate.setBackground(Color.CYAN);
btnCreate.setFont(new Font("",45,15));
btnDisplay = new JButton("Display");
btnDisplay.setForeground(Color.BLUE);
btnDisplay.setBackground(Color.CYAN);
btnClear = new JButton("Clear");
btnClear.setForeground(Color.BLUE);
btnClear.setBackground(Color.YELLOW);
```

```
btnGoDebit = new JButton("Go to Debit Card");
btnGoDebit.setForeground(Color.BLUE);
btnGoDebit.setBackground(Color.GREEN);
btnGoCredit = new JButton("Go to Credit Card");
btnGoCredit.setBackground(Color.GREEN);
btnGoCredit.setForeground(Color.BLUE);

lblTitle.setBounds(210,10,180,30);
lblID.setBounds(30,60,100,30);
txtID.setBounds(130,60,130,30);
lblAccount.setBounds(30,105,100,30);
txtAccount.setBounds(130,105,130,30);
lblName.setBounds(30,150,100,30);
txtName.setBounds(130,150,130,30);
lblAmount.setBounds(30,195,100,30);
txtAmount.setBounds(130,195,130,30);
lblIssuer.setBounds(330,60,100,30);
txtIssuer.setBounds(420,60,130,30);
lblPin.setBounds(330,105,100,30);
txtPin.setBounds(420,105,130,30);
lblConfirmPin.setBounds(330,150,100,30);
txtConfirmPin.setBounds(420,150,130,30);
lblDate.setBounds(330,195,100,30);
boxDD.setBounds(400,195,50,30);
```

```
boxMM.setBounds(450,195,50,30);

boxYY.setBounds(500,195,80,30);

btnCreate.setBounds(110,270,180,40);

btnDisplay.setBounds(300,270,180,40);

btnClear.setBounds(210,320,180,40);

btnGoDebit.setBounds(210,370,180,40);

btnGoCredit.setBounds(210,420,180,40);


fBankCard.add(pBankCard);

pBankCard.add(lblTitle);

pBankCard.add(lblID);

pBankCard.add(txtID);

pBankCard.add(lblAccount);

pBankCard.add(txtAccount);

pBankCard.add(lblName);

pBankCard.add(txtName);

pBankCard.add(lblAmount);

pBankCard.add(txtAmount);

pBankCard.add(lblIssuer);

pBankCard.add(txtIssuer);

pBankCard.add(lblPin);

pBankCard.add(txtPin);

pBankCard.add(lblConfirmPin);

pBankCard.add(txtConfirmPin);
```

```
pBankCard.add(lblDate);
pBankCard.add(boxDD);
pBankCard.add(boxMM);
pBankCard.add(boxYY);
pBankCard.add(btnCreate);
pBankCard.add(btnDisplay);
pBankCard.add(btnClear);
pBankCard.add(btnGoDebit);
pBankCard.add(btnGoCredit);

pBankCard.setLayout(null);
pBankCard.setSize(600,550);
pBankCard.setVisible(true);
fBankCard.setVisible(true);
fBankCard.setSize(600,550);
fBankCard.setResizable(false);
fBankCard.setLocationRelativeTo(null);
pBankCard.setBackground(Color.DARK_GRAY);
fBankCard.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

fDebitCard = new JFrame("Debit Card");
pDebitCard = new JPanel();
lblDebTitle = new JLabel("Debit Card");
lblDebTitle.setFont(new Font("",30,30));
```

```
lblDebTitle.setForeground(Color.CYAN);
lblDebID = new JLabel("Card ID:");
lblDebID.setForeground(Color.CYAN);
txtDebID = new JTextField();
txtDebID.setForeground(Color.BLUE);
txtDebID.setFont(new Font("",25,15));
btnDebGo = new JButton("Go");
btnDebGo.setBackground(Color.CYAN);
btnDebGo.setForeground(Color.BLUE);
lblDebAccount = new JLabel("Bank Account:");
lblDebAccount.setForeground(Color.CYAN);
txtDebAccount = new JTextField();
txtDebAccount.setForeground(Color.BLUE);
txtDebAccount.setFont(new Font("",25,15));
lblDebName = new JLabel("Client Name:");
lblDebName.setForeground(Color.CYAN);
txtDebName = new JTextField();
txtDebName.setForeground(Color.BLUE);
txtDebName.setFont(new Font("",25,15));
lblDebAmount = new JLabel("Balance Amount:");
lblDebAmount.setForeground(Color.CYAN);
txtDebAmount = new JTextField();
txtDebAmount.setForeground(Color.BLUE);
txtDebAmount.setFont(new Font("",25,15));
```

```
lblDebIssuer = new JLabel("Issuer Bank:");
lblDebIssuer.setForeground(Color.CYAN);
txtDebIssuer = new JTextField();
txtDebIssuer.setForeground(Color.BLUE);
txtDebIssuer.setFont(new Font("",25,15));
lblDebPin = new JLabel("Pin:");
lblDebPin.setForeground(Color.CYAN);
txtDebPin = new JTextField();
txtDebPin.setForeground(Color.BLUE);
txtDebPin.setFont(new Font("",25,15));
lblWithDraw = new JLabel("WithDraw:");
lblWithDraw.setForeground(Color.CYAN);
txtWithDraw = new JTextField();
txtWithDraw.setForeground(Color.BLUE);
txtWithDraw.setFont(new Font("",25,15));
lblDebDate = new JLabel("Date:");
lblDebDate.setForeground(Color.CYAN);
String Debday[] = {"1","2","3","4","5","6","7","8","9","10","11","12"};
boxDebDD = new JComboBox<String>(Debday);
boxDebDD.setEditable(true);
String Debmonth[] = {"1","2","3","4","5","6","7","8","9","10","11","12"};
boxDebMM = new JComboBox<String>(Debmonth);
boxDebMM.setEditable(true);
String Debyear[] = {"2023","2024","2025","2026"};
```

```
boxDebYY = new JComboBox<String>(Debyear);
boxDebYY.setEditable(true);
btnWithDraw = new JButton("WithDraw");
btnWithDraw.setBackground(Color.CYAN);
btnWithDraw.setForeground(Color.BLUE);
btnDebDisplay = new JButton("Display");
btnDebDisplay.setBackground(Color.CYAN);
btnDebDisplay.setForeground(Color.BLUE);
btnDebClear = new JButton("Clear");
btnDebClear.setBackground(Color.YELLOW);
btnDebClear.setForeground(Color.BLUE);
btnGoBank = new JButton("Go to Bank Card");
btnGoBank.setBackground(Color.GREEN);
btnGoBank.setForeground(Color.BLUE);

lblDebTitle.setBounds(210,5,180,30);
lblDebID.setBounds(30,60,100,30);
txtDebID.setBounds(130,60,100,30);
btnDebGo.setBounds(230,60,60,28);
lblDebAccount.setBounds(30,105,100,30);
txtDebAccount.setBounds(130,105,130,30);
lblDebName.setBounds(30,150,100,30);
txtDebName.setBounds(130,150,130,30);
lblDebAmount.setBounds(30,240,100,30);
```

```
txtDebAmount.setBounds(130,240,130,30);
lblDebIssuer.setBounds(30,195,100,30);
txtDebIssuer.setBounds(130,195,130,30);
lblWithDraw.setBounds(330,60,100,30);
txtWithDraw.setBounds(420,60,130,30);
lblDebPin.setBounds(330,105,100,30);
txtDebPin.setBounds(420,105,130,30);
lblDebDate.setBounds(330,150,100,30);
boxDebDD.setBounds(400,150,50,30);
boxDebMM.setBounds(450,150,50,30);
boxDebYY.setBounds(500,150,80,30);
btnWithDraw.setBounds(360,215,180,40);
btnDebDisplay.setBounds(360,270,180,40);
btnDebClear.setBounds(210,340,180,40);
btnGoBank.setBounds(210,390,180,40);

fDebitCard.add(pDebitCard);
pDebitCard.add(lblDebTitle);
pDebitCard.add(lblDebID);
pDebitCard.add(txtDebID);
pDebitCard.add(btnDebGo);
pDebitCard.add(lblDebAccount);
pDebitCard.add(txtDebAccount);
pDebitCard.add(lblDebName);
```

```
pDebitCard.add(txtDebName);
pDebitCard.add(lblDebIssuer);
pDebitCard.add(txtDebIssuer);
pDebitCard.add(lblDebAmount);
pDebitCard.add(txtDebAmount);
pDebitCard.add(lblDebPin);
pDebitCard.add(txtDebPin);
pDebitCard.add(lblWithDraw);
pDebitCard.add(txtWithDraw);
pDebitCard.add(lblDebDate);
pDebitCard.add(boxDebDD);
pDebitCard.add(boxDebMM);
pDebitCard.add(boxDebYY);
pDebitCard.add(btnWithDraw);
pDebitCard.add(btnDebDisplay);
pDebitCard.add(btnDebClear);
pDebitCard.add(btnGoBank);
pDebitCard.setLayout(null);
pDebitCard.setVisible(false);
pDebitCard.setSize(600,550);
pDebitCard.setBackground(Color.DARK_GRAY);

fDebitCard.setVisible(false);
fDebitCard.setResizable(false);
```

```
fDebitCard.setSize(600,550);
fDebitCard.setLocationRelativeTo(null);
fDebitCard.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);

fCreditCard = new JFrame("Credit Card");
pCreditCard = new JPanel();
lblCreTitle = new JLabel("Credit Card");
lblCreTitle.setForeground(Color.CYAN);
lblCreTitle.setFont(new Font("",30,30));
lblCreID = new JLabel("Card ID:");
lblCreID.setForeground(Color.CYAN);
txtCreID = new JTextField();
txtCreID.setForeground(Color.BLUE);
txtCreID.setFont(new Font("",25,15));
btnCreGo = new JButton("Go");
btnCreGo.setForeground(Color.BLUE);
btnCreGo.setBackground(Color.CYAN);
lblCreAccount = new JLabel("Bank Account:");
lblCreAccount.setForeground(Color.CYAN);
txtCreAccount = new JTextField();
txtCreAccount.setForeground(Color.BLUE);
txtCreAccount.setFont(new Font("",25,15));
lblCreName = new JLabel("Client Name:");
lblCreName.setForeground(Color.CYAN);
```

```
txtCreName = new JTextField();
txtCreName.setForeground(Color.BLUE);
txtCreName.setFont(new Font("",25,15));
lblCreAmount = new JLabel("Balance Amount:");
lblCreAmount.setForeground(Color.CYAN);
txtCreAmount = new JTextField();
txtCreAmount.setForeground(Color.BLUE);
txtCreAmount.setFont(new Font("",25,15));
lblCreIssuer = new JLabel("Issuer Bank:");
lblCreIssuer.setForeground(Color.CYAN);
txtCreIssuer = new JTextField();
txtCreIssuer.setForeground(Color.BLUE);
txtCreIssuer.setFont(new Font("",25,15));
lblCrePin = new JLabel("Pin:");
lblCrePin.setForeground(Color.CYAN);
txtCrePin = new JTextField();
txtCrePin.setForeground(Color.BLUE);
txtCrePin.setFont(new Font("",25,15));
lblCvc = new JLabel("CVC NO:");
lblCvc.setForeground(Color.CYAN);
txtCvc = new JTextField();
txtCvc.setForeground(Color.BLUE);
txtCvc.setFont(new Font("",25,15));
lblLimit = new JLabel("Credit Limit:");
```

```
lblLimit.setForeground(Color.CYAN);
txtLimit = new JTextField();
txtLimit.setForeground(Color.BLUE);
txtLimit.setFont(new Font("",25,15));
lblInterest = new JLabel("Interest Rate:");
lblInterest.setForeground(Color.CYAN);
txtInterest = new JTextField();
txtInterest.setForeground(Color.BLUE);
txtInterest.setFont(new Font("",25,15));
lblExpiration = new JLabel("Expiration:");
lblExpiration.setForeground(Color.CYAN);
//JTextField txtExpiration = new JTextField();
String Creday[] = {"1","2","3","4","5","6","7","8","9","10","11","12"};
boxCreDD = new JComboBox<String>(Creday);
boxCreDD.setEditable(true);
String Cremonth[] = {"1","2","3","4","5","6","7","8","9","10","11","12"};
boxCreMM = new JComboBox<String>(Cremonth);
boxCreMM.setEditable(true);
String Creyear[] = {"2023","2024","2025","2026"};
boxCreYY = new JComboBox<String>(Creyear);
boxCreYY.setEditable(true);
lblGrace = new JLabel("Grace Period:");
lblGrace.setForeground(Color.CYAN);
txtGrace = new JTextField();
```

```
txtGrace.setForeground(Color.BLUE);
txtGrace.setFont(new Font("",25,15));
btnCreAdd = new JButton("Add");
btnCreAdd.setForeground(Color.BLUE);
btnCreAdd.setBackground(Color.CYAN);
btnCreDisplay = new JButton("Display");
btnCreDisplay.setForeground(Color.BLUE);
btnCreDisplay.setBackground(Color.CYAN);
btnSet = new JButton("Set Credit Limit");
btnSet.setForeground(Color.BLUE);
btnSet.setBackground(Color.CYAN);
btnCancel = new JButton("Cancel Credit Card");
btnCancel.setForeground(Color.WHITE);
btnCancel.setBackground(Color.RED);
btnCreClear = new JButton("Clear");
btnCreClear.setForeground(Color.BLUE);
btnCreClear.setBackground(Color.YELLOW);
btnCreGoBank = new JButton("Go to Bank Card");
btnCreGoBank.setForeground(Color.BLUE);
btnCreGoBank.setBackground(Color.GREEN);

lblCreTitle.setBounds(210,5,180,30);
lblCreID.setBounds(30,60,100,30);
txtCreID.setBounds(130,60,100,30);
```

```
btnCreGo.setBounds(230,60,60,28);
lblCreAccount.setBounds(30,105,100,30);
txtCreAccount.setBounds(130,105,130,30);
lblCreName.setBounds(30,150,100,30);
txtCreName.setBounds(130,150,130,30);
lblCreAmount.setBounds(30,240,100,30);
txtCreAmount.setBounds(130,240,130,30);
lblCreIssuer.setBounds(30,195,100,30);
txtCreIssuer.setBounds(130,195,130,30);
lblCrePin.setBounds(30,285,100,30);
txtCrePin.setBounds(130,285,130,30);
lblCvc.setBounds(330,105,100,30);
txtCvc.setBounds(420,105,130,30);
lblLimit.setBounds(330,150,100,30);
txtLimit.setBounds(420,150,130,30);
lblInterest.setBounds(330,195,100,30);
txtInterest.setBounds(420,195,130,30);
lblExpiration.setBounds(330,285,100,30);
//txtExpiration.setBounds(420,285,130,30);
boxCreDD.setBounds(400,285,50,30);
boxCreMM.setBounds(450,285,50,30);
boxCreYY.setBounds(500,285,80,30);
lblGrace.setBounds(330,240,100,30);
txtGrace.setBounds(420,240,130,30);
```

```
btnCreAdd.setBounds(40,350,160,40);
btnSet.setBounds(210,350,160,40);
btnCancel.setBounds(380,350,160,40);
btnCreDisplay.setBounds(200,400,180,40);
btnCreClear.setBounds(200,450,180,40);
btnCreGoBank.setBounds(200,500,180,40);

fCreditCard.add(pCreditCard);
pCreditCard.add(lblCreTitle);
pCreditCard.add(lblCreID);
pCreditCard.add(txtCreID);
pCreditCard.add(lblCreAccount);
pCreditCard.add(txtCreAccount);
pCreditCard.add(lblCreName);
pCreditCard.add(txtCreName);
pCreditCard.add(lblCreIssuer);
pCreditCard.add(txtCreIssuer);
pCreditCard.add(lblCreAmount);
pCreditCard.add(txtCreAmount);
pCreditCard.add(lblCrePin);
pCreditCard.add(txtCrePin);
pCreditCard.add(lblCvc);
pCreditCard.add(txtCvc);
pCreditCard.add(lblLimit);
```

```
pCreditCard.add(txtLimit);
pCreditCard.add(lblInterest);
pCreditCard.add(txtInterest);
pCreditCard.add(lblExpiration);
//pCreditCard.add(txtExpiration);
pCreditCard.add(boxCreDD);
pCreditCard.add(boxCreMM);
pCreditCard.add(boxCreYY);
pCreditCard.add(lblGrace);
pCreditCard.add(txtGrace);
pCreditCard.add(btnCreGo);
pCreditCard.add(btnCreAdd);
pCreditCard.add(btnCreDisplay);
pCreditCard.add(btnSet);
pCreditCard.add(btnCancel);
pCreditCard.add(btnCreClear);
pCreditCard.add(btnCreGoBank);
pCreditCard.setLayout(null);
pCreditCard.setVisible(false);
pCreditCard.setSize(600,600);
pCreditCard.setBackground(Color.DARK_GRAY);

fCreditCard.setVisible(false);
fCreditCard.setResizable(false);
```

```
        fCreditCard.setSize(600,600);
        fCreditCard.setLocationRelativeTo(null);
        fCreditCard.getContentPane().setBackground(Color.CYAN);
        fCreditCard.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);

        fDisplay=new JFrame("Display");
        pDisplay=new JPanel();
        JTextArea textArea=new JTextArea();
        textArea.setEditable ( false ); // set textArea non-editable
        JScrollPane scroll = new JScrollPane ( textArea );
        scroll.setVerticalScrollBarPolicy                                    (
ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED );
        GridLayout layout1=new GridLayout(1,1);
        pDisplay.setLayout(layout1);
        pDisplay.add(scroll);
        pDisplay.setSize(400,500);
        fDisplay.add(pDisplay);
        fDisplay.setSize(400,500);
        fDisplay.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        fDisplay.setVisible(false);

        btnGoDebit.addActionListener(new ActionListener()
          {
             public void actionPerformed(ActionEvent e)
             {
```

```
            fDebitCard.setVisible(true);
            pDebitCard.setVisible(true);
            fBankCard.setVisible(false);
            pBankCard.setVisible(false);
        }
    });

    btnGoBank.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            fDebitCard.setVisible(false);
            pDebitCard.setVisible(false);
            fBankCard.setVisible(true);
            pBankCard.setVisible(true);
        }
    });

    btnCreGoBank.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            fBankCard.setVisible(true);
            pBankCard.setVisible(true);
```

```
            fCreditCard.setVisible(false);
            pCreditCard.setVisible(false);
        }
    });

    btnGoCredit.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            fBankCard.setVisible(false);
            pBankCard.setVisible(false);
            fCreditCard.setVisible(true);
            pCreditCard.setVisible(true);
        }
    });

    btnClear.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            txtID.setText("");
            txtAccount.setText("");
            txtName.setText("");
            txtAmount.setText("");
```

```
            txtIssuer.setText("");
            txtPin.setText("");
            txtConfirmPin.setText("");
        }
    });

    btnDebClear.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            txtDebID.setText("");
            txtDebAccount.setText("");
            txtDebName.setText("");
            txtDebAmount.setText("");
            txtDebIssuer.setText("");
            txtDebPin.setText("");
            txtWithDraw.setText("");
        }
    });

    btnCreClear.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
```

```
            txtCreID.setText("");
            txtCreAccount.setText("");
            txtCreName.setText("");
            txtCreAmount.setText("");
            txtCreIssuer.setText("");
            txtCrePin.setText("");
            txtCvc.setText("");
            txtLimit.setText("");
            txtInterest.setText("");
            txtGrace.setText("");
        }
    });

btnCreate.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            boolean check = true;
            for(BankCard cards : bankcard)
            {
                if(cards instanceof BankCard)
                {
                    BankCard bankcard = (BankCard) cards;
                    if(txtID.getText().equals(String.valueOf(bankcard.getcardID())))
```

```
                    {
                        check = false;
                        break;
                    }
                    else
                    {
                        check = true;
                    }
                }
            }
            try
            {
                if(check == true)
                {
                    if(Integer.parseInt(txtID.getText()) < 0
||txtName.getText().isEmpty() || txtAmount.getText().isEmpty() ||
Integer.parseInt(txtAmount.getText()) < 0
                    || txtID.getText().isEmpty() || txtAccount.getText().isEmpty() ||
txtIssuer.getText().isEmpty() || txtPin.getText().isEmpty() ||
Integer.parseInt(txtPin.getText()) < 0)
                    {
                        JOptionPane.showMessageDialog(null,"Input
Invalid","Error",JOptionPane.ERROR_MESSAGE);
                    }
                    else
                    {
```

```
                    if(txtPin.getText().equals(txtConfirmPin.getText()))
                    {
                        int id = Integer.parseInt(txtID.getText());
                        String account = txtAccount.getText();
                        String name = txtName.getText();
                        double balance = Double.parseDouble(txtAmount.getText());
                        String bank = txtIssuer.getText();
                        int pin = Integer.parseInt(txtPin.getText());

                        BankCard objbankcard = new BankCard(balance,id,account,bank);
                        DebitCard debitcard = new DebitCard(balance,id,account,bank,name,pin);
                        bankcard.add(debitcard);
                        JOptionPane.showMessageDialog(null,"Account Created!","Message",JOptionPane.WARNING_MESSAGE);

                    }

                    else
                    {
                        JOptionPane.showMessageDialog(null,"Check Pin","Error",JOptionPane.ERROR_MESSAGE);
                    }
                }
```

```
                }
                else if(check == false)
                {
                    JOptionPane.showMessageDialog(null,"Card ID already Use!","Error",JOptionPane.ERROR_MESSAGE);
                }
            }
            catch(NumberFormatException ex)
            {
                JOptionPane.showMessageDialog(null,"Check All the Data CareFully","Error",JOptionPane.ERROR_MESSAGE);
            }
        }

    });

    btnDisplay.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            fDisplay.setVisible(true);
            StringBuilder builder = new StringBuilder();
            String day = String.valueOf(boxDD.getSelectedItem());
            String month = String.valueOf(boxMM.getSelectedItem());
            String year = String.valueOf(boxYY.getSelectedItem());
```

```
            for(BankCard card : bankcard)
            {
                if(card instanceof DebitCard)
                {
                    DebitCard objDebitCard = (DebitCard) card;
                    builder.append("Card ID:" +objDebitCard.getcardID() +"\n");
                    builder.append("Bank                                    Account:"
+objDebitCard.getBankAccount() +"\n");
                    builder.append("Client  Name:"  +objDebitCard.getClientName()
+"\n");
                    builder.append("Balance
Amount:"+objDebitCard.getBalanceAmount() +"\n");
                    builder.append("Issuer   Bank:"   +objDebitCard.getissureBank()
+"\n");
                    builder.append("Pin:" +objDebitCard.getPinNumber() +"\n");
                    builder.append("Date:" +day+"/"+month+"/"+year+"\n");
                    builder.append("-------------------------------------------------" +"\n");
                }
            }
            if(builder.toString().isEmpty())
            {
                textArea.setText("No Data Found");
            }
            else
            {
```

```
                textArea.setText(builder.toString());
            }
        }
    });

    btnDebGo.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            boolean check = true;
            try
            {
                int ids = Integer.parseInt(txtDebID.getText());
                for(BankCard cards: bankcard){
                    if(cards instanceof DebitCard)
                    {

                        DebitCard debitcard = (DebitCard) cards;
                        if( ids == debitcard.getcardID())
                        {
                            check = true;
                            break;
                        }
                        else
```

```
                {
                    check = false;
                }

            }

        }

        if(check == true)
        {
            for(BankCard card : bankcard)
            {
                if(card instanceof DebitCard)
                {

                    //int id = Integer.parseInt(txtDebID.getText());

                    DebitCard objDebitCard = (DebitCard) card;

                    if(ids == objDebitCard.getcardID())
                    {

txtDebAccount.setText(objDebitCard.getBankAccount());
                        txtDebAccount.setEditable(false);
                        txtDebName.setText(objDebitCard.getClientName());
```

```
                        txtDebName.setEditable(false);
                        txtDebIssuer.setText(objDebitCard.getissureBank());
                        txtDebIssuer.setEditable(false);

txtDebAmount.setText(String.valueOf(objDebitCard.getBalanceAmount()));
                        txtDebAmount.setEditable(false);
                    }
                }
            }
            }
            if(check == false)
            {
                JOptionPane.showMessageDialog(null,"Card ID Not
Found","Error",JOptionPane.ERROR_MESSAGE);
            }
        }
        catch(NumberFormatException ex)
        {
            JOptionPane.showMessageDialog(null,"Check All the Data
CareFully","Error",JOptionPane.ERROR_MESSAGE);
        }
    }
});

    btnWithDraw.addActionListener(new ActionListener()
```

```
        {
            public void actionPerformed(ActionEvent e)
            {
                try
                {
                    for(BankCard card : bankcard)
                    {
                        if(card instanceof DebitCard)
                        {

                            int withdraw = Integer.parseInt(txtWithDraw.getText());
                            int pin = Integer.parseInt(txtDebPin.getText());
                            String day = String.valueOf(boxDebDD.getSelectedItem());
                            String month = String.valueOf(boxDebMM.getSelectedItem());
                            String year = String.valueOf(boxDebYY.getSelectedItem());
                            String date = String.valueOf(day+"/"+month+"/"+year);

                            DebitCard objDebitCard = (DebitCard) card;

if(txtDebID.getText().equals(String.valueOf(objDebitCard.getcardID())))
                            {

if(txtDebPin.getText().equals(String.valueOf(objDebitCard.getPinNumber())))
                                {
                                    if(Integer.parseInt(txtWithDraw.getText()) < 0)
```

```
                    {
                        JOptionPane.showMessageDialog(null,"Amount
cannot be Negative!","Error",JOptionPane.ERROR_MESSAGE);
                    }
                    else
                    {
                        if(Integer.parseInt(txtWithDraw.getText())<=
objDebitCard.getBalanceAmount())
                        {
                            //objDebitCard.setWithdrawalAmount(withdraw);
                            objDebitCard.withdraw(withdraw,date,pin);

//objDebitCard.setBalanceAmount(objDebitCard.getBalanceAmount()      -
objDebitCard.getWithdrawalAmount());
                            JOptionPane.showMessageDialog(null,"Card
ID:"+objDebitCard.getcardID()+"\n"+"Bank                    Account:"
+objDebitCard.getBankAccount()+"\n"+
                                    "Client
Name:"+objDebitCard.getClientName()+"\n"+                    "Issuer
Bank:"+objDebitCard.getissureBank()+"\n"+"Pin:"+objDebitCard.getPinNumber()+
"\n"+
                                    "Balance
Amount:"+objDebitCard.getBalanceAmount()+"\n"+"WithDrawal
Amount:"+objDebitCard.getWithdrawalAmount()+"\n"+"Date                Of
WithDrawal:"+objDebitCard.getDateOfWithdrawal()
                                    +"\n"+"Amount
WithDrawed","Message",JOptionPane.INFORMATION_MESSAGE);
```

```
                                }
                                else
                                {
                                    JOptionPane.showMessageDialog(null,"Insufficent
Balance","Error",JOptionPane.ERROR_MESSAGE);
                                }
                            }
                        }
                        else
                        {
                            JOptionPane.showMessageDialog(null,"Incorrect
Pin!","Error",JOptionPane.ERROR_MESSAGE);
                        }
                    }


                }
            }
        }
        catch(NumberFormatException ex)
        {
            JOptionPane.showMessageDialog(null,"Check Data
CareFully","Error",JOptionPane.ERROR_MESSAGE);
        }
    }
});
```

```
btnDebDisplay.addActionListener(new ActionListener()
  {
    public void actionPerformed(ActionEvent e)
    {
      fDisplay.setVisible(true);
      StringBuilder builder = new StringBuilder();
      for(BankCard card : bankcard)
      {
        if(card instanceof DebitCard)
        {
          DebitCard objDebitCard = (DebitCard) card;
          builder.append("Card ID:" +objDebitCard.getcardID() +"\n");
          builder.append("Bank Account:"
+objDebitCard.getBankAccount() +"\n");
          builder.append("Client Name:" +objDebitCard.getClientName()
+"\n");
          builder.append("Balance
Amount:"+objDebitCard.getBalanceAmount() +"\n");
          builder.append("Issuer Bank:" +objDebitCard.getissureBank()
+"\n");
          builder.append("Pin:" +objDebitCard.getPinNumber() +"\n");
          builder.append("WithDrawal Amount:"
+objDebitCard.getWithdrawalAmount() +"\n");
          builder.append("Date of WithDrawal:"
+objDebitCard.getDateOfWithdrawal() +"\n");
```

```
                    builder.append("Has WithDrawan:"
+objDebitCard.getHasWithdrawn() +"\n");
                    builder.append("-----------------------------------------------" +"\n");
                }
            }
            if(builder.toString().isEmpty())
            {
                textArea.setText("No Data Found");
            }
            else
            {
                textArea.setText(builder.toString());
            }
        }
    });

    btnCreGo.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            boolean check = true;
            for(BankCard cards : bankcard)
            {
                if(cards instanceof BankCard)
                {
```

```
                BankCard bankcard = (BankCard) cards;

if(txtCreID.getText().equals(String.valueOf(bankcard.getcardID())))
                {
                    check = true;
                    break;
                }
                else
                {
                    check = false;
                }
            }
        }

        if(check == true)
        {
            for(BankCard card : bankcard)
            {
                if(card instanceof BankCard)
                {
                    BankCard objBankCard = (BankCard) card;
                    if(Integer.parseInt(txtCreID.getText())          ==
objBankCard.getcardID())
                    {
                        txtCreAccount.setText(objBankCard.getBankAccount());
```

```
                    txtCreAccount.setEditable(false);
                    txtCreName.setText(objBankCard.getClientName());
                    txtCreName.setEditable(false);
                    txtCreIssuer.setText(objBankCard.getissureBank());
                    txtCreIssuer.setEditable(false);

txtCreAmount.setText(String.valueOf(objBankCard.getBalanceAmount()));
                    txtCreAmount.setEditable(false);
                }
            }
        }
    }
    else
    {
        JOptionPane.showMessageDialog(null,"Card ID Not
Found","Error",JOptionPane.ERROR_MESSAGE);
    }
  }
});

  btnCreAdd.addActionListener(new ActionListener()
  {
    public void actionPerformed(ActionEvent e)
    {
      try{
```

```
                int id = Integer.parseInt(txtCreID.getText());
                String account = txtCreAccount.getText();
                String name = txtCreName.getText();
                double balance = Double.parseDouble(txtCreAmount.getText());
                String bank = txtCreIssuer.getText();
                int cvc = Integer.parseInt(txtCvc.getText());
                double rate = Double.parseDouble(txtInterest.getText());
                String day = String.valueOf(boxCreDD.getSelectedItem());
                String month = String.valueOf(boxCreMM.getSelectedItem());
                String year = String.valueOf(boxCreYY.getSelectedItem());
                String date = String.valueOf(day+"/"+month+"/"+year);

                CreditCard objCreditCard = new
CreditCard(id,name,bank,account,balance,cvc, rate,date);
                bankcard.add(objCreditCard);
                JOptionPane.showMessageDialog(null,"CreditCard
Added","Message",JOptionPane.INFORMATION_MESSAGE);
            }
            catch(NumberFormatException ex)
            {
                JOptionPane.showMessageDialog(null,"Check Data
Carefully!","Error",JOptionPane.ERROR_MESSAGE);
            }
        }
    });
```

```
btnSet.addActionListener(new ActionListener()
  {
    public void actionPerformed(ActionEvent e)
    {
      for(BankCard card: bankcard)
      {
        if(card instanceof CreditCard)
        {
          try{
            CreditCard objCreditCard = (CreditCard) card;
            double limit = Double.parseDouble(txtLimit.getText());
            int grace = Integer.parseInt(txtGrace.getText());

if(txtCreID.getText().equals(String.valueOf(objCreditCard.getcardID())))
            {
              if(objCreditCard.getIsGranted() == false)
              {
                if(Double.parseDouble(txtLimit.getText()) <= 2.5 *
objCreditCard.getBalanceAmount())
                {
                  objCreditCard.setCreditLimit(limit,grace);
                  JOptionPane.showMessageDialog(null,"Card
ID:"+objCreditCard.getcardID()+"\n"+"Credit
Limit:"+objCreditCard.getCreditLimit()+"\n"
```

```
                                    +"Grace
Period:"+objCreditCard.getGracePeriod()+"\n"+"Credit
Issued","Message",JOptionPane.INFORMATION_MESSAGE);
                                }
                                else
                                {
                                    JOptionPane.showMessageDialog(null,"Credit cannot
be Issued","Error",JOptionPane.ERROR_MESSAGE);
                                }
                            }
                            else
                            {
                                JOptionPane.showMessageDialog(null,"Credit  Already
taken","Error",JOptionPane.ERROR_MESSAGE);
                            }
                        }
                    }
                    catch(NumberFormatException ex)
                    {
                        JOptionPane.showMessageDialog(null,"Check  Data
CareFully","Error",JOptionPane.ERROR_MESSAGE);
                    }
                }
            }
        }
    });
```

```
btnCancel.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            String x =JOptionPane.showInputDialog(null,"Enter your pin","Question",JOptionPane.QUESTION_MESSAGE);
            for(BankCard card : bankcard)
            {
                if(card instanceof CreditCard)
                {
                    CreditCard objCreditCard = (CreditCard) card;

if(txtCreID.getText().equals(String.valueOf(objCreditCard.getcardID())))
                    {
                        if(x.equals(txtCrePin.getText()))
                        {
                            if(objCreditCard.getIsGranted() == true)
                            {
                                objCreditCard.cancelCreditCard();
                                JOptionPane.showMessageDialog(null,"Credit Cancelled","Message",JOptionPane.INFORMATION_MESSAGE);
                            }
                            else
                            {
```

```
                    JOptionPane.showMessageDialog(null,"Credit Not
Taken Yet!","Error",JOptionPane.ERROR_MESSAGE);
                }
            }
            else
            {
                JOptionPane.showMessageDialog(null,"Pin
Incorrect","Error",JOptionPane.ERROR_MESSAGE);
            }
        }
    }
}
});

    btnCreDisplay.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            fDisplay.setVisible(true);
            StringBuilder builder = new StringBuilder();
            for(BankCard card : bankcard)
            {
                if(card instanceof CreditCard)
                {
```

```
                    CreditCard objCreditCard = (CreditCard) card;
                    builder.append("Card ID:" +objCreditCard.getcardID() +"\n");
                    builder.append("Bank Account:"
+objCreditCard.getBankAccount() +"\n");
                    builder.append("Client Name:" +objCreditCard.getClientName()
+"\n");
                    builder.append("Balance
Amount:"+objCreditCard.getBalanceAmount() +"\n");
                    builder.append("Issuer Bank:" +objCreditCard.getissureBank()
+"\n");
                    builder.append("CVC Number:"
+objCreditCard.getCVC_Number() +"\n");
                    builder.append("Grace Period:"
+objCreditCard.getGracePeriod() +"\n");
                    builder.append("Credit Limit:" +objCreditCard.getCreditLimit()+
"\n");
                    builder.append("Interest Rate:"
+objCreditCard.getInterestRate() +"\n");
                    builder.append("Expiration Date:"
+objCreditCard.getExpirationDate() +"\n");
                    builder.append("-------------------------------------------------" +"\n");
                }
            }
            if(builder.toString().isEmpty())
            {
                textArea.setText("No Data Found");
            }
```

```
                else
                {
                    textArea.setText(builder.toString());
                }
            }
        });
    }

    public static void main(String []args)
    {
        new BankGUI();
    }
}
```